University of South Carolina
Scholar Commons

Theses and Dissertations

2015

Trees, Partitions, and Other Combinatorial Structures

Heather Christina Smith
University of South Carolina

Follow this and additional works at: https://scholarcommons.sc.edu/etd

Part of the Mathematics Commons

Recommended Citation

Smith, H. C.(2015). *Trees, Partitions, and Other Combinatorial Structures.* (Doctoral dissertation). Retrieved from https://scholarcommons.sc.edu/etd/3678

This Open Access Dissertation is brought to you by Scholar Commons. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of Scholar Commons. For more information, please contact digres@mailbox.sc.edu.

Trees, Partitions, and Other Combinatorial Structures

by

Heather Christina Smith

Bachelor of Arts
Houghton College 2008

Master of Science
Virginia Commonwealth University 2010

Submitted in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy in

Mathematics

College of Arts and Sciences

University of South Carolina

2015

Accepted by:

László A. Székely, Major Professor

Éva Czabarka, Committee Member

Jerrold R. Griggs, Committee Member

George F. McNulty, Committee Member

Csilla Farkas, Committee Member

Lacy K. Ford, Jr., Senior Vice Provost and Dean of Graduate Studies

© Copyright by Heather Christina Smith, 2015
All Rights Reserved.

Dedication

This dissertation is dedicated to my grandmother, Peggy H. Smith.

Acknowledgments

First and foremost, László Székely, thank you for your guidance throughout my time at the University of South Carolina. You have taught me many things about mathematics research, sharing exciting problems as well as equipping me with useful tools and techniques. You have challenged me as a young researcher to be a life-long learner. You helped me write my first grant proposal and introduced me to several of your collaborators so that I could begin projects with them as well. You have given me insight into every aspect of the profession amidst fun stories. You have well prepared me for a career in academia. László, thank you for investing in me.

To Éva Czabarka, thank you for the many ways in which you have advised me for my future career. You have been a wonderful resource as I have sought to understand the biological models behind my research. I have enjoyed working with you on several open problems. You are an excellent teacher inside and outside the classroom, taking the time to explain background material and paint a big picture when the details become tedious. Éva, thank you for always having time to answer my questions.

Thank you, George McNulty, for introducing me to universal algebras. It has been an exciting journey starting with the first year algebra sequence, through courses for my comprehensive exams, and each Friday in the Algebra & Logic Seminar. I have learned so much from you and am excited to carry these tools with me to my next position.

Jerry Griggs, I always looked forward to my first year Discrete course. With a bit of humor in each lesson, you made the material exciting. Many of the bonus problems you posed will stay in my files for future exploration. In studying for your

comprehensive exam, I learned wealth of combinatorial tools and structures, many of which have been useful in my own research. Thank you for laying a foundation for me in Discrete mathematics.

Thank you, Csilla Farkas, for taking the time to serve on my committee. Steve Fenner, thank you for your help with the computational complexity topics. To the many other faculty, staff, and graduate students in the mathematics departments, thank you for your part in helping me to mature as a mathematician as well as for your encouragement and support.

István Miklós, thank you for working with me these past couple years. The projects have inspired me and significantly shaped my research program. They have opened doors for me regarding the next steps in my career. I am eager to continue working with you.

Hua Wang, it has been a pleasure to work with you these past two years and I look forward to cracking open new projects with you in the future. Thank you for your diligence and patience. You are a model for me on how to maintain a long-distance collaboration.

Finally, I would like to thank my family for their love and support. You have always been there for me with encouragement to pursue my dreams and never give up. Thank you for teaching me to keep my focus on the Lord, without whom I could not have come this far. Thank you for showing me that serving Him is more rewarding than any degree or position that I could hold.

The work in this dissertation was supported in part by the following grants:

- Contract #FA9550-12-1-0405 of the U.S. Air Force Office of Scientific Research (AFOSR) and the Defense Advanced Research Projects Agency (DARPA).
- Contract #1300547 of the NSF DMS.

- Dean's Doctoral Dissertation Fellowship from the College of Arts and Sciences of the University of South Carolina.
- SPARC Graduate Fellowship from the Office of the Vice President for Research at the University of South Carolina.
- Simons Foundation (#245307).

Abstract

This dissertation contains work on three main topics.

Chapters 1 through 4 provide complexity results for the single cut-or-join model for genome rearrangement. Genomes will be represented by binary strings. Let $\mathcal{S}$ be a finite collection of binary strings, each of the same length. Define $\mathcal{M}$ to be the collection of medians – binary strings μ which minimize $\sum_{\nu \in S} H(\mu, \nu)$ where H is the Hamming distance. For any non-negative function $f(x)$, define $Z(f(x), \mathcal{S})$ to be $\sum_{\mu \in \mathcal{M}} \prod_{\nu \in \mathcal{S}} f(H(\mu, \nu))$. We study the complexity of calculating $Z(f(x), \mathcal{S})$, with respect to the number of strings in $\mathcal{S}$ and their length.

If the leaves of a star are labeled with the strings in $\mathcal{S}$, then $Z(x!, \mathcal{S})$ counts the pairs of functions where one selects a median μ for $\mathcal{S}$ and the other assigns, to each $\nu \in \mathcal{S}$, a permutation of coordinates in which μ and ν differ. This relates to the small parsimony problem for genome rearrangement. We show that it is #P-complete to calculate $Z(x!, \mathcal{S})$ and give similar results for other functions $f(x)$. We also consider an analogous problem when the leaves of a binary tree are labeled. This is joint work with István Miklós.

Chapters 5 and 6 explore tree invariants. In particular, Chapter 5 examines the eccentricity of a vertex, $\mathrm{ecc}_T(v) = \max_{u \in T} d_T(v, u)$ where $d_T(u, v)$ is the number of edges along the path connecting u and v in T. This was one of the first, distance-based, tree invariants studied (Jordan 1869). The total eccentricity of a tree, $\mathrm{Ecc}(T)$, is the sum of the eccentricities of its vertices. We determine extremal values and characterize extremal tree structures for the ratios $\mathrm{Ecc}(T)/\,\mathrm{ecc}_T(u)$, $\mathrm{Ecc}(T)/\,\mathrm{ecc}_T(v)$, $\mathrm{ecc}_T(u)/\,\mathrm{ecc}_T(v)$, and $\mathrm{ecc}_T(u)/\,\mathrm{ecc}_T(w)$ where u, w are leaves of T and v is in the

center of T. Analogous problems have been resolved for other tree invariants including distance (Barefoot, Entringer, and Székely 1997) and the number of subtrees (Székely and Wang 2013). In addition, we determine the tree structures that minimize and maximize total eccentricity among trees with a given degree sequence. This is joint work with László Székely and Hua Wang.

Chapter 6 compares three different middle parts of a tree. Different middle parts such as center, centroid, subtree core have been defined and studied. We want to provide some general insights on the difference between them and consider how far apart (with given order of the tree) two different 'middle point' can be and when such maximum distances are achieved. This study, after conducted on general trees, is naturally extended to trees with restricted degrees or diameter due to the evident correlation between these restrictions and the maximum distance between middle parts. Some related interesting questions arise that may be of interest independently. This is joint work with László Székely, Hua Wang, and Shuai Yuan.

Chapter 7 studies a problem related to Baranyai's Theorem. This guarantees that whenever k divides n, there is a partition of $\binom{[n]}{k}$ into rows such that each row is itself a partition of $[n]$. Baranyai (1973) used graph flows to give an existence proof for this 118 year old conjecture. We are interested in the structure of these partitions. For $k = 2$, there is a circular configuration which yields a straightforward construction. Beth (1974) found an algebraic construction for $k = 3$. However neither method has a known extension to larger k. We consider a new construction for $k = 2$ which makes use of a bijection between partitions and labeled trees. It is our hope that this type of connection will lead to a more general construction of Baranyai partitions. This is joint work with László Székely.

Table of Contents

Dedication iii

Acknowledgments iv

Abstract vii

List of Tables xi

List of Figures xii

Chapter 1 Genome Rearrangement 1

1.1 Mathematical Model 7

1.2 Computational complexity 10

Chapter 2 Result for Star Phylogenetic Trees 16

2.1 Encoding a clause 16

2.2 Complexity result for #StarSPSCJ 28

2.3 Torpidly mixing Markov chain 35

Chapter 3 Generalizations for the Star Tree 42

3.1 Calculating #StarSPSCJ(f) exactly 42

3.2 Stochastic Approximations 61

Chapter 4 Result for Binary Phylogenetic Trees 68

4.1 Algorithms for finding most parsimonious labelings 68

4.2 Complexity result for #BinSPSCJ 76

Chapter 5 Eccentricity Sums in Trees 90

5.1 Extremal ratios 91

5.2 Extremal structures 101

Chapter 6 On different "middle parts" of a tree 113

6.1 Definitions and Characterizations 113

6.2 Maximum distances between middle parts in general trees 119

6.3 Trees with degree restrictions 127

6.4 Different "middle parts" in trees with a given diameter D 138

6.5 Rooted trees of given order and height 141

Chapter 7 Some Remarks on Baranyai's Theorem 157

7.1 Baranyai's Theorem 158

7.2 Bijection with binary trees 158

7.3 Tree construction 159

7.4 New $(n, 2)-$Baranyai partitions 165

7.5 Conclusion 170

Bibliography 171

Appendix A Method for Selecting the Binary Strings in $\mathcal{C}_i$ . . . 175

List of Tables

Table 2.1 The 50 strings in $\mathcal{C}_i$ for a single clause c_i along with their Hamming distance from medians in $\mathcal{M}'$. 22

Table 2.2 A key for interpreting Column A of Table 2.1. 24

Table 3.1 The 26 strings to complement the collection in Table 2.1 along with their Hamming distance from medians in $\mathcal{M}'$. 59

Table 5.1 A summary of results for an arbitrary tree T on n vertices with $v \in C(T)$ and $u, w \in L(T)$. 101

Table A.1 Hamming distances when $\eta \in N_1$ with $\eta[x_\alpha] = 0$ and $\eta[y_\alpha] = 1$. . . 178

Table A.2 The 8 different sets of 3 strings in N_2 and their Hamming distances with medians in $\mathcal{M}'$. 182

List of Figures

Figure 1.1 Three representations of a genome. The binary string representation can be obtaining by reading the rows of the array. 3

Figure 1.2 Two genomes with 6 possible SCJ scenarios between them. . . . 5

Figure 1.3 Two genomes with 4! = 24 different SCJ scenarios that will transform $\mathcal{G}$ into $\mathcal{G}'$. The adjacency graph $A(\{\mathcal{G}, \mathcal{G}'\})$ shows that $\{\mathcal{G}, \mathcal{G}'\}$ has independent adjacencies. 6

Figure 4.1 Comb connecting copies of $\mathcal{U}_i$ to create $\mathcal{T}_i$. 78

Figure 4.2 The labeled binary tree on the right is $S(a, b, c)$. The representation on the left will be used in place of $S(a, b, c)$ in future figures. 79

Figure 4.3 The binary tree $\hat{\mathcal{B}}_i$, for $i \in [k]$ created for clause $c_i = x_1 \vee x_2 \vee x_3$. 80

Figure 5.1 A tree which minimizes $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)}$. 93

Figure 5.2 A tree (left) which maximizes $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)}$ and a tree (right) which minimizes $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)}$. 95

Figure 5.3 Trees which minimize $\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(v)}$, the right one for even n only. 99

Figure 5.4 A tree which maximizes $\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(w)}$. 100

Figure 5.5 Non-isomorphic greedy caterpillars for a given degree sequence . . 103

Figure 5.6 Generating T' from T for the proof of Proposition 5.11. 104

Figure 5.7 A greedy tree. 106

Figure 5.8 A level-greedy tree. 106

Figure 5.9 A tree rooted at v with daughter subtree, T_1, containing leaves of height h. 107

Figure 5.10 Creating $T_{\pi''}$ from $T_{\pi'}$ for the proof of Theorem 5.21. 111

Figure 6.1 A tree with $v \in C(T)$, $u \in CT(T)$, $w \in Core(T)$ which do not lie on a common path. 115

Figure 6.2 An example of trees with vertices from the center, centroid, and subtree core on a common path, but in different orders. 116

Figure 6.3 An r-comet of order n. 119

Figure 6.4 A tree T with $u \in CT(T)$, $v \in C(T)$, $w \in L(T)$, and all vertices not in T_u are on the path $P(u, w)$. 120

Figure 6.5 A representation of tree T for the proof of Theorem 6.11 with path $P(u, v)$, T_u, T_v, and w labeled. 124

Figure 6.6 A rooted greedy tree with given degree sequence and root degree 2. 128

Figure 6.7 An extremal binary tree which is conjectured to maximize the distance between each pair of middle sets. 130

Figure 6.8 An extended good tree with 33 vertices and maximum degree 4. . 131

Figure 6.9 An extended rgood tree with 29 vertices and maximum degree 4. 132

Figure 6.10 The structure of a tree T with diameter D and order n which maximize $d(Core(T), CT(T))$. 140

Figure 6.11 Trees $T(x)$ and $T'(x)$ from Lemma 6.32 142

Figure 6.12 Trees $T(u)$ and $T'(u)$ in the proof of Lemma 6.33. 143

Figure 6.13 Transforming $T(x)$ into $T'(x)$ when $deg_T(v) = 3$ in the proof of Lemma 6.35. 146

Figure 6.14 Transforming $T(x)$ into $T'(x)$ when v has degree 4 in the proof of Lemma 6.35. 147

Figure 6.15 Trees S and S' from the proof of Lemma 6.38 149

Figure 6.16 Trees S and S' from the proof of Lemma 6.39. 150

Figure 6.17 The structure of a tree T with height h and order n which minimizes the number of root-containing subtrees. 153

Figure 7.1 One-to-one correspondence between leaf-labeled binary trees and 2-partitions. 159

Figure 7.2 The single binary tree for the $(2, 2)$-Baranyai partition. 160

Figure 7.3 Creating a tree in $\mathcal{T}_4'$ from a tree in $\mathcal{T}_4$. 161

Figure 7.4 The general caterpillar representing matching 35|26|47|18. 162

Figure 7.5 Creating a tree in $\mathcal{T}_6'$ from a tree in $\mathcal{T}_6$. 163

Figure 7.6 A caterpillar representation of the extension of the matching $\{3, x\}, \{1, y\}$. 164

Figure 7.7 A circular representation to find an $(8, 2)$-Baranyai partitions. . . 166

Figure 7.8 Illustration of the deductions in Case 1 of Claim 7.7 167

Figure 7.9 Illustration of the deductions in Case 2/Red of Claim 7.7 168

Figure 7.10 Illustration of the contradiction to c_1 being blue in Case 2/Red of Claim 7.7. 168

Figure 7.11 Illustration of the deductions made when c_1 is red in Case 2/Red of Claim 7.7. 169

Figure 7.12 The set-up for Case 2/Blue. 169

Figure 7.13 Illustration of deductions made in Case 2/Blue of Claim 7.7. . . . 169

Figure 7.14 Illustration of the deductions made when c_1 is blue in Case 2/Blue of Claim 7.7. 170

Figure A.1 A labeling of the 3-dimensional cube with the possible values of $\mu[S_i]$. 177

Figure A.2 Display of $H(\eta_j[S_i], \mu[S_i])$ on the cube for $\eta_\alpha, \eta_\beta, \eta_\gamma \in N_1$. 179

Figure A.3 The left cube displays the values of $H_1(\mu)$ and the right cube displays the values of $\overline{H_1}(\mu)$. 180

Figure A.4 The values $\left\{H(\eta[S_i], \mu[S_i]) : \eta \in \mathcal{N}_1^{(+3)}\right\} \uplus \overline{H_1}(\mu)$ displayed on the median cube. 180

Figure A.5 The left cube displays the Hamming distances $H(\mu[S_i], \zeta_{\alpha,\beta}[S_i])$. The middle gives $H_2(\mu)$ and the right cube displays $\overline{H_2}(\mu)$. 181

Figure A.6 Hamming distances with a string τ with $N(\tau) = 3$. 183

Figure A.7 A selection which distinguishes the Hamming distances at u_f from the rest. 184

Chapter 1

Genome Rearrangement

Soon after Sturtvant (1913) developed the first genetic map, he published his observations (Sturtvant 1921) regarding genome rearrangement in the fruit fly *Drosophila melanogaster*. Later Dobzhansky and Sturtvant (1938) took a deeper look at genome rearrangement, analyzing the rearrangement scenarios for two species, *Drosophila pseudoobscura* and *Drosophila miranda*. Palmer and Herbon (1988) focused their studies on genome rearrangement in plants and began the discussion about most parsimonious scenarios. We build upon this foundation, exploring the Single Cut-or-Join model for genome rearrangement.

A *genome* is represented by an edge-labeled digraph where each vertex has total degree (sum of in degree and out degree) at most two. Each directed edge represents a *gene* (or synteny block). Each gene has a head and a tail, collectively called *extremities*. Vertices of degree two are called *adjacencies* while vertices of degree one are called *telomeres*. Because of the degree restriction, each extremity can participate in at most one adjacency.

There are many ways to represent genomes, each giving a different viewpoint of the structure. Here we give a representation of genomes through their adjacency graphs. Fix a set of genes $\{v_1, v_2, \ldots, v_m\}$. Let h_{v_i} denote the head of gene v_i and t_{v_i} denote the tail. For a genome $\mathcal{G}$ on this set of genes, create an adjacency graph $A(\mathcal{G})$ where the vertices are precisely the extremities of the genes in (G):

$$V(A(\mathcal{G})) = \{h_{v_1}, t_{v_1}, h_{v_2}, t_{v_2}, \ldots, h_{v_m}, t_{v_m}\}.$$

Connect two extremities with an edge if they form an adjacency in $\mathcal{G}$.

While each edge in $A(\mathcal{G})$ represents an adjacency in $\mathcal{G}$, each vertex of $A(\mathcal{G})$ with degree zero indicates a telomere. Consequently, we may use unordered pairs of gene extremities to describe an adjacency and a single gene extremity to identify a telomere. Because each extremity participates in at most one adjacency, the degree of each vertex is at most one and $A(\mathcal{G})$ must be a matching (possibly empty or perfect). Observe that the adjacency graph $A(\mathcal{G})$ is uniquely defined by $\mathcal{G}$.

Given a set of m genes, create a graph A with

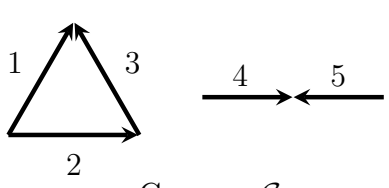
$$V(A) = \{h_{v_1}, t_{v_1}, h_{v_2}, t_{v_2}, \ldots, h_{v_m}, t_{v_m}\}$$

and an edge set which is a matching. This graph will uniquely describe a genome on the given genes. It is then evident that genomes can be uniquely described by their set of genes and their adjacencies, for any extremity which does not appear in an adjacency must be a telomere.

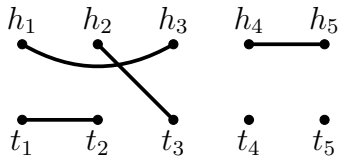
Next we explore the interaction of multiple genomes on the same set of genes. Fix a set of m genes $\{v_1, v_2, \ldots, v_m\}$ and a multiset of n genomes $\mathbb{G} = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_n\}$ on the same set of specified genes. Draw a single adjacency graph $A(\mathbb{G})$, where $V(A(\mathbb{G}))$ is the set of gene extremities and an edge is drawn between two extremities if there is at least one genome in $\mathbb{G}$ which has the corresponding adjacency. We say that $\mathbb{G}$ has *independent adjacencies* if $E(A)$ is a matching for A.

The final characterization makes use of binary strings. Fix a set of m genes and a genome $\mathcal{G}$ on these genes. Define a binary string with $\binom{2m}{2}$ coordinates. The coordinates are in one-to-one correspondence with the possible adjacencies (pairs of gene extremities). The genome $\mathcal{G}$ is then represented by the binary string which has a 1 in each coordinate that corresponds to an adjacency in $\mathcal{G}$ and a 0 in every other coordinate.

The binary string representation seems to be the most concise representation, but it is more difficult to verify the vertex degree condition for $\mathcal{G}$. However, given a set of m genes and a multiset of genomes $\mathbb{G}$ on those genes which have independent



Genome $\mathcal{G}$



Adjacency graph $A(\mathcal{G})$

Binary String Representation of $\mathcal{G}$

	h_1	t_1	h_2	t_2	h_3	t_3	h_4	t_4	h_5	t_5
h_1		0	0	0	1	0	0	0	0	0
t_1			0	1	0	0	0	0	0	0
h_2				0	0	1	0	0	0	0
t_2					0	0	0	0	0	0
h_3						0	0	0	0	0
t_3							0	0	0	0
h_4								0	1	0
t_4									0	0
h_5										0
t_5										

Figure 1.1 Three representations of a genome. The binary string representation can be obtaining by reading the rows of the array.

adjacencies, the binary string representation is preferred. In this setting, the number of different adjacencies that appear in at least one genome in $\mathbb{G}$ is m, so we can restrict our binary strings at most m coordinates. The usefulness of this representation will become more evidence after we discuss the Single Cut-or-Join model for genome rearrangement.

From here forward, every multiset of genomes will consist of genomes on the same set of genes. For a multiset of genomes observed in current species, we use a tree to represent their phylogenetic history, labeling the leaves of the tree with the given genomes. The most ancient vertex in the tree can be considered the root, a common ancestor of the genomes in the leaves. Each internal vertex represents an unknown species. Given a fixed tree and a labeling of its leaves with genomes, we seek the most likely genomes with which to label these internal nodes, according to some criterion. In particular, we use the parsimony criterion to determine likelihood. Before this, we first define a model for genome rearrangement.

Along each edge of the tree, subsequent mutations occurred to transform the genome of the ancestor into the genome of its descendant. To describe these mutations, we use the Single Cut-or-Join model for genome rearrangement.

Definition 1.1 (Feijão and Meidanis (2011))**.** *A* Single Cut-or-Join (SCJ) *operation transforms one genome into another genome by altering the set of adjacencies in exactly one of the following ways:*

- *Cut: replace adjacency* $u = \{x, y\}$ *with telomeres* $u_1 = \{x\}$ *and* $u_2 = \{y\}$*;*
- *Join: replace telomeres* $u_1 = \{x\}$ *and* $u_2 = \{y\}$ *with the single adjacency* $u = \{x, y\}$.

The parsimony principle asserts that the true phylogenetic history minimizes the number of mutations that must take place along the edges of the tree. Feijão and Meidanis (2011) showed that the minimum number of SCJ operations needed to transform one genome into another is precisely the Hamming distance between their binary string representations. This is achieved by first cutting all of the adjacencies in the ancestor's genome which do not occur in the descendant's genome. Then make the necessary joins to obtain the genome of the descendant. As we consider the possible ancestors which could label the internal nodes of the tree, we use the parsimony score to determine the likelihood of a labeling. For a tree T and labeling φ of the vertices of T with binary strings, the parsimony score is precisely the sum of the Hamming distances between labelings on adjacent vertices; symbolically,

$$\sum_{uv \in E(T)} H(\varphi(u), \varphi(v))$$

where $H(.,.)$ is the Hamming distance between the two inputs. The labelings with minimum parsimony score are called "most parsimonious labelings" and are considered to be the most likely.

Figure 1.2 Two genomes with 6 possible SCJ scenarios between them.

Given two genomes on the same set of genes, in the process of one changing into the other, we assume that only one SCJ operation happens at a time because mutations are generally rare events. A time-order sequence of these SCJ operations, transforming one genome into another with the fewest number of SCJ operations is an *SCJ scenario.* We may note that the number of SCJ scenarios between two genomes is at most the factorial of the Hamming distance between their binary strings. In practice, there may be fewer SCJ scenarios because the digraph produced from each subsequent cut or join must be a valid genome.

Example 1.2. *For the two genomes in Figure 1.2, the number of SCJ operations needed to transform $\mathcal{G}$ into $\mathcal{G}'$ is 4 because $\mathcal{G}$ has two adjacencies which are not in $\mathcal{G}'$ ($\{h_2, t_3\}, \{t_1, h_4\}$) and $\mathcal{G}'$ also has two adjacencies not in $\mathcal{G}$ ($\{h_2, t_5\}, \{t_5, h_4\}$). While there are two cuts and two joins that must be made, notice that the first SCJ operation must be a cut.*

An SCJ scenario for $\mathcal{G}$ and $\mathcal{G}'$ must be either follow a cut-cut-join-join pattern (4 different SCJ scenarios) or an alternating cut-join-cut-join pattern (2 different SCJ scenarios). If, for example, the first SCJ operation was to cut the adjacency $\{h_2, t_3\}$, then the second SCJ operation could be to join $\{h_2\}$ and $\{t_5\}$, but it could not be to join $\{h_4\}$ and $\{h_5\}$ because h_4 is not a telomere yet. As a result, there are there are precisely 6 possible SCJ scenarios that will transform $\mathcal{G}$ into $\mathcal{G}'$.

When we restrict our attention to a pair of genomes, $\{\mathcal{G}, \mathcal{G}'\}$, with independent adjacencies, then the vertices of an adjacency $\{x, y\}$ that appears in $\mathcal{G}'$ but not in $\mathcal{G}$

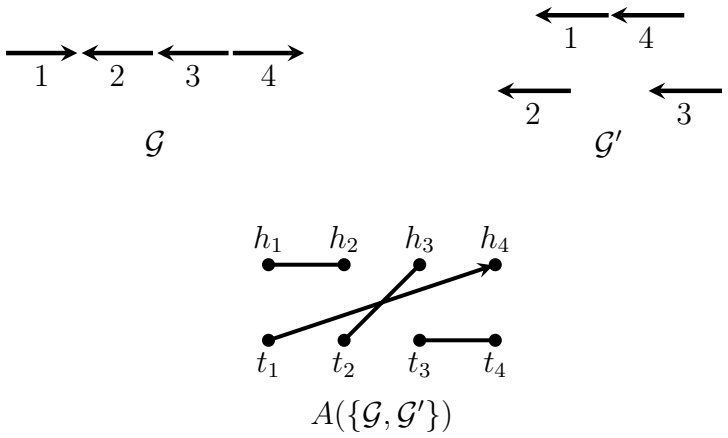


Figure 1.3 Two genomes with $4! = 24$ different SCJ scenarios that will transform $\mathcal{G}$ into $\mathcal{G}'$. The adjacency graph $A(\{\mathcal{G}, \mathcal{G}'\})$ shows that $\{\mathcal{G}, \mathcal{G}'\}$ has independent adjacencies.

must appear in two telomeres, $\{x\}$ and $\{y\}$ in $\mathcal{G}$. The same holds for an adjacency which appears in $\mathcal{G}$ but not in $\mathcal{G}'$. Therefore, among the set of cuts and joins that must be performed to transform $\mathcal{G}$ into $\mathcal{G}'$, performing one necessary SCJ operation will not affect the ability to make a different SCJ operation. Therefore, any ordering is possible and the number of possible SCJ scenarios is the factorial of the Hamming distance between the binary string representations of $\mathcal{G}$ and $\mathcal{G}'$.

Example 1.3. *Figure 1.3 defines two genomes such that the adjacencies of* $\{\mathcal{G}, \mathcal{G}'\}$ *are independent as indicated by the adjacency graph* $A(\{\mathcal{G}, \mathcal{G}'\})$.

To transform $\mathcal{G}$ *into* $\mathcal{G}'$*, there are 3 cuts and 1 join that must be made. Because of independence, these operations can be performed in any order. As a result, there are* $4! = 24$ *different SCJ scenarios.*

On the phylogenetic tree endowed with a most parsimonious labeling of the vertices, we will assign to each edge an SCJ scenario which details the transformation of the ancestral genome at one endpoint to the descendant genome labeling its other endpoint. We make the assumption that the SCJ scenario on one edge has no influence over the SCJ scenario on a different edge.

All of these definitions will be made precise in the next section. Here is a brief summary of the structures we will be counting. We are given a multiset of genomes, all on the same set of genes, and an ancestral tree T relating them. These will be viewed as a tree T with binary strings labeling the leaves. The first goal is to determine the possible most parsimonious labelings of the internal nodes of T. For each most parsimonious labeling, we then label each edge of the tree with an SCJ scenario which details the time order that mutations occurred in transforming the genome at one endpoint into the genome at the other. The result is a most parsimonious SCJ scenario.

1.1 Mathematical Model

Now let us fix some notation and formalize a few of the definitions which were introduced above. Throughout the paper, we let $[m] := \{1, 2, \ldots, m\}$. For two binary strings η and η' of the same length, we let $H(\eta, \eta')$ be the Hamming distance between η and η', which is the number of coordinates in which η and η' differ. Every multiset, S, of binary strings will have the property that no two strings in S differ in length. For an arbitrary binary string η of length n and coordinates labeled by the integers 1 through n, we will use the notation $\eta[z]$ to denote the value of η in the z coordinate for $z \in [n]$. For a multiset A, we define $\#[x, A]$ to be the multiplicity of the element x in A. (If $x \notin A$ then $\#[x, A] = 0$.) To explicitly write out a multiset, we use subscripts in parentheses to indicate the multiplicity of an element. For example,

$$\{a_{(3)}, b_{(4)}, c_{(1)}\} := \{a, a, a, b, b, b, b, c\}.$$

For two multisets A and B, we define $A \uplus B$ to be the multiset with

$$\#[x, A \uplus B] = \#[x, A] + \#[x, B].$$

Definition 1.4 (Most Parsimonious Labeling)**.** *Let T be a tree with s leaves. Let $B = \{v_1, v_2, \ldots, v_s\}$ be a multiset of binary strings which represent genomes with m*

genes. Let $\phi : L(T) \to B$ *be a surjection which assigns a binary string to each leaf of* T*. A* most parsimonious labeling *of the vertices of* T*, endowed with leaf-labeling* ϕ*, is a labeling* $\phi' : V(T) \to \{0,1\}^{\binom{2t}{2}}$ *with*

- $\phi'(\ell) = \phi(\ell)$ *for each* $\ell \in L(T)$ *(i.e.* ϕ' *extends* ϕ*),*
- $\phi'(v)$ *corresponds to a valid genome for each* $v \in V(T)$*, and*
- $\sum_{uv \in E(T)} H(\phi'(u), \phi'(v))$ *is minimum among the possible functions* ϕ'*.*

Definition 1.5 (SCJ scenarios)**.** *Let* η *and* η' *be two genomes on the same set of* m *genes. An* SCJ scenario *for the pair* (η, η') *is a minimum length sequence of cuts and joins to be performed consecutively in order to transform* η *into* η' *with the condition that each subsequent cut or join creates a valid genome.*

In the binary string representation, an SCJ scenario is just a permutation of the coordinates in which the binary strings differ, telling the order in which the bits should be flipped. If the adjacencies are independent, then any permutation is an SCJ scenario. Otherwise, one must verify that the binary string resulting from each subsequent bit-flip creates a valid genome.

Let T *be a tree with most parsimonious labeling* ϕ'*. For* $uv \in E(T)$*, an SCJ scenario for* $(\phi'(u), \phi'(v))$ *will also be called an SCJ scenario for the edge* uv*.*

Definition 1.6. *Let* B *be a multiset of genomes on the same set of genes. Draw the adjacency graph* $A(B)$ *as described in Section 1. If the edges form a matching, then we say that the adjacencies of multiset* B *are* independent*.*

Note that if $\{\eta, \eta'\}$ has independent adjacencies, then the number of different SCJ scenarios for (η, η') is precisely the factorial of the Hamming distance between their binary string representations.

Definition 1.7 (Most parsimonious SCJ scenario)**.** *Let* T *be a tree and* ϕ *be a function labeling the leaves of* T *with binary strings. A* most parsimonious SCJ scenario *for* T

and ϕ consists of a most parsimonious labeling ϕ' and an SCJ scenario for each edge of T under vertex labeling ϕ'.

Definition 1.8. *Given a tree T and leaf labeling ϕ, let ϕ' be a most parsimonious labeling. We say that the* number of SCJ scenarios admitted by *ϕ' is precisely the number of ways to assign an SCJ scenario to each edge of T.*

When the tree T is a star with leaf labeling ϕ, each most parsimonious labeling ϕ' of the vertices is characterized by the binary string μ it assigns to the center vertex of the star. We call μ a median. Formally, we define a median as follows:

Definition 1.9 (Median)**.** *Let $B = \{\nu_i\}_{i=1}^m$ be a multiset of binary strings which represent a multiset of genomes on the same m genes. A binary string μ which represents a genome on these m genes and minimizes $\sum_{i\in[m]} H(\nu_i, \mu)$ is called a* median *for B.*

If the multiset of genomes has independent adjacencies or if a multiset of binary strings is given without reference to genomes, then any binary string which minimizes $\sum_{i\in[m]} H(\nu_i, \mu)$ is a median.

The next definition will fit into the context of our later proofs. Here we merely state the definition.

Definition 1.10. *For an arbitrary $n, t \in \mathbb{Z}^+$, let B be an arbitrary multiset of binary strings of length $2n + t$. We use $\mathcal{M}(B)$ to denote the set of all medians for B. If the strings of B are defined on the coordinates*

$$(x_1, y_1, x_2, y_2, \ldots, x_n, y_n, e_1, e_2, \ldots, e_t),$$

we use $\mathcal{M}'(B)$ to denote the subset of $\mathcal{M}(B)$ containing only those medians μ with $\mu[x_i] \neq \mu[y_i]$ for all $i \in [n]$.

While it is possible for $\mathcal{M}'(B)$, in the previous definition, to be empty, when we use the definition later, most of our multisets B are defined so that $\mathcal{M} = \{0,1\}^{2n} \times \{0\}^t$ and $\mathcal{M} = \{01,10\}^n \times \{0\}^t$.

1.2 Computational complexity

While P and NP are complexity classes for decision problems, the following classes are for counting problems.

The classes #P, #P-hard, and #P-complete were first defined by Valiant (1979). The definition for #P that we give here, while not the original, is an equivalent definition.

Definition 1.11 (Welsh (1993))**.** *The class* #P *contains those functions* $f : \Sigma^* \to \mathbb{N}$*, for some alphabet* Σ*, such that both of the following hold:*

- *There is a polynomial* p*, a relation* R*, and a polynomial time algorithm which, for each input* $w \in \Sigma^*$ *and each* $y \in \Sigma^*$ *with* $|y| \leq p(|w|)$*, determines if* $R(w, y)$*.*
- *For any input* w*,* $f(w) = |\{y : |y| \leq p(|w|) \text{ and } R(w, y)\}|$*.*

Definition 1.12 (Cook (1971))**.** *A polynomial time reduction from one decision (counting) problem* A *to another decision (counting) problem* B *is an algorithm which, for an arbitrary instance of* A*,*

- *runs in time polynomial in the inputs of the instance of* A*,*
- *creates an instance of* B*,*
- *the answer to each instance of* A *can be computed in polynomial time from the answer of the instance of* B*.*

Definition 1.13 (Valiant (1979))**.** *A counting problem is in* #P-hard *if there is a polynomial time reduction to it from every problem in #P. A counting problem is in #P-complete if it is in #P and is in #P-hard.*

Next we give a few known computational complexity results. To state these result, we establish some terminology.

Closed normal form (CNF) is a standard format in which to express Boolean formulas. A *3CNF* is a Boolean formula Γ which is the conjunction of clauses and each clause is the disjunction of 3 literals. The symbol $\wedge$ is used for conjunction and the symbol $\vee$ is for disjunction. A 3CNF, Γ, with n variables $\{v_1, v_2, \ldots, v_n\}$ and k clauses takes the form $\Gamma = c_1 \wedge c_2 \wedge \ldots \wedge c_k$ where each c_i is a clause which is the disjunction of three literals and the literals are from $\{v_i\}_{i=1}^n \cup \{\overline{v_i}\}_{i=1}^n$. Because Γ was said to have n variables, we may assume that, for each $i \in [n]$, v_i or $\overline{v_i}$ appears in some clause of Γ. Each v_i is a *positive literal* while each $\overline{v_i}$ is a *negative literal*. The negative literal $\overline{v_i}$ is the negation of v_i. We identify $\overline{\overline{v_i}}$ with the literal v_i. We refer to $\{v_i\}_{i=1}^m$ as the *variables* of Γ and always assume that the set of variables has an ordering.

A *truth assignment* for Γ is a function $f : \{v_i\}_{i=1}^n \to \{T, F\}$ which assigns a value of true or false to each variable. If a truth assignment makes Γ true, we say it satisfies Γ. Otherwise, a truth assignment does not satisfy Γ in which case there is at least one clause which is not satisfied.

Definition 1.14 (3SAT)**.** *Given an arbitrary Γ in 3CNF with n variables and k clauses, decide if there is a truth assignment for Γ which satisfies Γ.*

Definition 1.15 (#3SAT)**.** *Given an arbitrary Boolean formula Γ in 3CNF with n variables and k clauses, count the number of truth assignments which satisfy Γ.*

Theorem 1.16 (Cook (1971))**.** *3SAT $\in$ NP-complete.*

Theorem 1.17 (Cook (1971))**.** *#3SAT $\in$ #P-complete.*

Define *D3CNF* to be the subset of 3CNF containing only those $\Gamma = \bigwedge_{i\in[k]} c_i$ such that for each $i \in [k]$,

- c_i contains three distinct literals, and
- c_i does not contain both v_j and $\overline{v_j}$ for any $j \in [n]$.

This defines the following two problems.

Definition 1.18 (D3SAT)**.** *For an arbitrary Γ in D3CNF with n variables and k clauses, decide if there a truth assignment which satisfies Γ.*

Definition 1.19 (#D3SAT)**.** *For an arbitrary Γ in D3CNF with n variables and k clauses, count the number of truth assignments which satisfy Γ.*

The following two results are proven through reductions from #3SAT and 3SAT.

Lemma 1.20. *#D3SAT $\in$ #P-complete.*

Proof. This is a reduction from #3SAT. Let Γ be a 3CNF with n variables and k clauses, $n \geq 3$. Let $v_\alpha, v_\beta, v_\gamma$ be literals in Γ with $\alpha \neq \beta \neq \gamma \neq \alpha$. Observe that each of the following pairs have the same satisfying truth assignments.

$(v_\alpha \vee v_\beta \vee v_\beta)$ and $(v_\alpha \vee v_\beta \vee v_\gamma) \wedge (v_\alpha \vee v_\beta \vee \overline{v_\gamma})$.

$(v_\alpha \vee v_\alpha \vee v_\alpha)$ and $(v_\alpha \vee v_\beta \vee v_\gamma) \wedge (v_\alpha \vee \overline{v_\beta} \vee v_\gamma) \wedge (v_\alpha \vee v_\beta \vee \overline{v_\gamma}) \wedge (v_\alpha \vee \overline{v_\beta} \vee \overline{v_\gamma})$.

Further, a clause of the form $(v_\alpha \vee \overline{v_\alpha} \vee v_\beta)$ is alway true, so it can be removed.

Making these replacements in Γ will result in a D3CNF Γ' with n' variables ($n' \leq n$) and at most $4k$ clauses. Because some clauses like $(v_\alpha \vee \overline{v_\alpha} \vee v_\beta)$ are in Γ but not in Γ', it is possible that $n' < n$.

Given a satisfying truth assignment for Γ', we may extend it to a satisfying truth assignment for Γ in $2^{n-n'}$ ways. This is because the variables in Γ which are not in Γ' do not affect the ability of a truth assignment to satisfy Γ. On the other hand,

each satisfying truth assignment for Γ, restricted to the variables of Γ', will be a satisfying truth assignment for Γ'. ∎

Lemma 1.21. *D3SAT ∈ NP-complete.*

Proof. As described in the last proof, for any 3CNF Γ, there is a D3CNF Γ' which is computable in polynomial time such that Γ' has at least one satisfying truth assignment exactly when Γ has at least one satisfying truth assignment. ∎

Next, we return our attention to the phylogenetic histories which were introduced in Section 1 and formalized in Section 1.1. The complexity results of these next 3 chapters address subquestions and analogues of the following problems.

Definition 1.22 (#SPSCJ)**.** *Given a tree T and a labeling φ of the leaves of T with binary strings, #SPSCJ asks for the exact number of most parsimonious SCJ scenarios.*

Lemma 1.23. *#SPSCJ ∈ #P.*

Proof. The input includes a tree T with n vertices, and a function $\varphi : L(T) \to \{0,1\}^\ell$. A witness is a function $\varphi' : V(T) \to \{0,1\}^\ell$ and a function which assigns an SCJ scenario (a permutation of a subset of $[\ell]$) to each edge of the tree. The size of the input is at most $O(\ell n)$.

Feijão and Meidanis (2011) gave a polynomial time algorithm to find one most parsimonious labeling. The parsimony score for this labeling can be calculated in time polynomial in the number of edges of T and in the length, ℓ, of the binary strings labeling the leaves of T. Then we need only compare this parsimony score with the parsimony score of the possible witness. If they are the same, then φ' is a most parsimonious labeling. For each edge, we can verify that a permutation assigned to an edge is an appropriate SCJ scenario for that edge in $O(\ell)$ time. By Definition 1.11, #SPSCJ is in #P. ∎

The next two definitions are for special cases of #SPSCJ. Notice that they are stated without mention of genes and genomes. The problems are modeling the case when a multiset of genomes has independent adjacencies. Since each labeling of ancestral genomes is most parsimonious, the genomes assigned to internal vertices will only contain subsets of the adjacencies appearing in the given multiset of genomes. Therefore, we restrict our binary string representations to these coordinates. Because the multiset of genomes has independent adjacencies, no two adjacencies being considered share a common extremity. Therefore, a binary string, under this coordinate restriction, may have all ones.

When the tree T is a star and the multiset of genomes labeling the leaves has independent adjacencies (Definition 1.6), we can state the following special case of #SPSCJ:

Definition 1.24 (#StarSPSCJ)**.** *Given an arbitrary $m \in \mathbb{Z}^+$, let $B = \{\nu_i\}_{i=1}^m$ be an arbitrary multiset of binary strings. Determine the value of*

$$\sum_{\mu \in \mathcal{M}(B)} \prod_{i \in [m]} H(\nu_i, \mu)!.$$

When the tree T is a binary tree and the genomes labeling the leaves have independent adjacencies (Definition 1.6), we state one more special case of #SPSCJ:

Definition 1.25 (#BinSPSCJ)**.** *Given arbitrary integer $m \geq 2$, let T be a binary tree with m leaves. Let $B = \{\nu_i\}_{i=1}^m$ be an arbitrary multiset of binary strings and a surjective function $\varphi : L(T) \to B$. Define F to be the set of most parsimonious labelings φ' which extend φ to $V(T)$. Determine the value of*

$$\sum_{\varphi' \in F} \prod_{uv \in E(T)} H(\varphi'(u), \varphi'(v))!.$$

The results in the next three chapters examine #SPSCJ and some analogues for classes of trees such as binary trees and star trees. Most of the complexity results are reductions from #D3SAT. In other words, given a D3CNF Γ with n variables and k clauses, we create a multiset of m binary strings of length $2n + t$ (where t and m are polynomials of n and k) to label the leaves of the tree. These strings will be chosen so that the number of most parsimonious SCJ scenarios is related to the number of satisfying truth assignments for Γ.

Chapter 2

Result for Star Phylogenetic Trees

This chapter examines the computational complexity of #StarSPSPCJ. The first section details some tools and constructions that will be needed for the proof of our main result in Section 2.2. This main result, Theorem 2.18, states that #StarSPSCJ is #P-complete.

2.1 Encoding a clause

The proof of Theorem 2.18 will define a polynomial reduction from #D3SAT (Definition 1.19) to #StarSPSCJ (Definition 1.24). Fix an arbitrary D3CNF, Γ, with n variables and k clauses. Fix a prime $p \leq 5\max\{300, n+5\}$ which will be utilized later. We will define a multiset of binary strings $\mathcal{D}(p)$ to label the leaves of a star. This multiset will encode Γ. Once the leaves of the star tree are labeled with our binary strings, we have an instance of #StarSPSCJ. For this instance, #StarSPSCJ asks us to count the number of most parsimonious SCJ scenarios. To do this, first fix a most parsimonious labeling and count the number of SCJ scenarios it admits (Definitions 1.4 and 1.8). Then sum this quantity over all most parsimonious labelings.

For the star, each most parsimonious labeling is identified by the median (Definition 1.9) which it assigns to the center of the star. As in Definition 1.10, we let $\mathcal{M}(\mathcal{D}(p))$ be the set of all medians for the star tree with leaf labels $\mathcal{D}(p)$.

Our task is to define a multiset of binary strings $\mathcal{D}(p)$ to encode Γ. The multiset $\mathcal{D}(p)$ will be chosen so that $\mathcal{M}(\mathcal{D}(p))$ will have a set of desired characteristics. First, each of our strings in $\mathcal{D}(p)$ and the medians $\mathcal{M}(\mathcal{D}(p))$ will have length $2n+t$ with

coordinates

$$(x_1, y_1, x_2, y_2, \ldots, x_n, y_n, e_1, e_2, \ldots, e_t)$$

where n is the number of variables in Γ and the t is a polynomial of n and k which will be defined later. Second, $\mathcal{M}(\mathcal{D}(p))$ will be the set of all binary strings μ of length $2n+t$ that have $\mu[e_i] = 0$ for each $i \in [t]$. In other words, $\mathcal{D}(p)$ will be defined so that $\mathcal{M}(\mathcal{D}(p))$ equals $\{0,1\}^{2n} \times \{0\}^t$. Recall $\mathcal{M}'(\mathcal{D}(p))$, from Definition 1.10, is the subset of $\mathcal{M}(\mathcal{D}(p))$ with the additional property that $\mu[x_i] \neq \mu[y_i]$ for all $i \in [n]$. Once we have established that $\mathcal{M}(\mathcal{D}(p)) = \{0,1\}^{2n} \times \{0\}^t$, we can conclude $\mathcal{M}'(\mathcal{D}(p)) = \{01, 10\}^n \times \{0\}^t$. This allows for a connection with truth assignments for Γ.

Definition 2.1. *Let $n \in \mathbb{Z}^+$. For arbitrary Γ in D3CNF with n variables, let S be a multiset of binary strings on the coordinates $(x_1, y_1, \ldots, x_n, y_n, e_1, \ldots, e_t)$. There is an injective function f which assigns to each median $\mu \in \mathcal{M}'(S)$ a truth assignment for Γ. In particular, $f(\mu)$ will assign a value of true to the i^{th} variable of Γ if $\mu[x_i] = 1$ and false if $\mu[x_i] = 0$.*

Remark 2.2. *If multiset S is chosen so that $\mathcal{M}'(S) = \{01, 10\}^n \times \{0\}^t$, then Definition 2.1 provides a bijection between $\mathcal{M}'(S)$ and the truth assignments for Γ.*

Definition 2.3. *Let $n \in \mathbb{Z}^+$. Given an arbitrary D3CNF, Γ, with n variables, let S be an arbitrary multiset of binary strings on the coordinates $(x_1, y_1, \ldots, x_n, y_n, e_1, \ldots, e_t)$ for some $t \in \mathbb{Z}^+$. Define $\mathcal{M}'_\Gamma(S)$ to be a subset of $\mathcal{M}'(S)$, containing only those medians which, through the bijection in Definition 2.1, correspond to a satisfying truth assignment for Γ. Since a single clause c in Γ is also a D3CNF, this defines $\mathcal{M}'_c(S)$ as well.*

For #StarSPSCJ, we are asked to calculate

$$\sum_{\mu \in \mathcal{M}(\mathcal{D}(p))} \prod_{i \in [m]} H(\mu, \nu_i)!.$$

To do this, we first calculate $\prod_{i\in[m]} H(\mu,\nu_i)!$ for each median $\mu \in \mathcal{M}(\mathcal{D}(p))$ where the product is the number of scenarios admitted by median μ (Definition 1.8). The multiset $\mathcal{D}(p)$ will be constructed so that there is a constant $K(p)$ (specified in Claim 2.22) which is with each string in $\mathcal{M}'_\Gamma(\mathcal{D}(p))$ admitting exactly $K(p)$ scenarios, and $K(p) \not\equiv 0 \mod p$. Each string in $\mathcal{M}(\mathcal{D}(p)) \setminus \mathcal{M}'_\Gamma(\mathcal{D}(p))$ will admit more or fewer than $K(p)$ scenarios, and further $\prod_{i\in[m]} H(\mu,\nu_i)! \equiv 0 \mod p$. As a result

$$\sum_{\mu\in\mathcal{M}(\mathcal{D}(p))} \prod_{i\in[m]} H(\mu,\nu_i)! \equiv |\mathcal{M}'_\Gamma(\mathcal{D}(p))|K(p) \mod p.$$

Repeating this construction for sufficiently many primes $p \leq 5\max\{300, n+5\}$, we obtain enough congruences, which together with the knowledge that there are at most 2^n satisfying truth assignment for Γ, uniquely determine the size of $\mathcal{M}'_\Gamma(\mathcal{D}(p))$ which is equal to the number of satisfying truth assignments for Γ.

Later we will see that the main work goes into developing a multiset $\mathcal{D}(p)$ with the property that $\mu \in \mathcal{M}'_\Gamma(\mathcal{D}(p))$ and $\mu' \in \mathcal{M}'(\mathcal{D}(p)) \setminus \mathcal{M}'_\Gamma(\mathcal{D}(p))$ have

$$\prod_{i\in[m]} H(\mu,\nu_i)! \neq \prod_{i\in[m]} H(\mu',\nu_i)!.$$

In Section 2.1.1, we define the strings $\mathcal{D}(p)$ which are used in the proofs to distinguish $\mathcal{M}'_\Gamma(\mathcal{D}(p))$ from $\mathcal{M}'(\mathcal{D}(p)) \setminus \mathcal{M}'_\Gamma(\mathcal{D}(p))$.

2.1.1 Encoding Boolean clauses in binary strings.

A truth assignment satisfies Γ if and only if it satisfies every clause in Γ. Hence, we will encode each clause c_i of Γ in a set of 50 strings

$$\mathcal{C}_i := \{\nu_1^i, \nu_2^i, \ldots, \nu_{50}^i\}$$

which will be defined through Table 2.1. These 50 strings, a subset of $\mathcal{D}(\Gamma)$, are designed to distinguish those medians in $\mathcal{M}'_{c_i}(\mathcal{C}_i)$ from those in $\mathcal{M}'(\mathcal{C}_i) \setminus \mathcal{M}'_{c_i}(\mathcal{C}_i)$. Confirmation of this will come in Section 2.1.3. Because every truth assignment

which does not satisfy Γ does not satisfy some clause in Γ, we will see that $\biguplus_{i\in[k]} \mathcal{C}_i$ distinguishes between $\mathcal{M}'_\Gamma\left(\biguplus_{i\in[k]} \mathcal{C}_i\right)$ and $\mathcal{M}'\left(\biguplus_{i\in[k]} \mathcal{C}_i\right) \setminus \mathcal{M}'_\Gamma\left(\biguplus_{i\in[k]} \mathcal{C}_i\right)$.

The following definition gives a guide for defining a multiset of binary strings.

Definition 2.4 (Defining strings)**.** *For arbitrary $m, n \in \mathbb{Z}^+$ and $t \in \mathbb{Z}^{\geq 0}$, to define a multiset of binary strings $\{\eta_1, \eta_2, \ldots, \eta_m\}$ on coordinates $(x_1, y_1, \ldots, x_n, y_n, e_1, \ldots, e_t)$, it suffices to*

- *define $\eta_j[x_\ell]$ and $\eta_j[y_\ell]$ for each $j \in [m]$ and $\ell \in [n]$, and*
- *define a function $e : [m] \to \mathbb{Z}^{\geq 0}$.*

We say η_j has $e(j)$ additional ones. In order to infer the values $\eta_j[e_\ell]$ for each $j \in [m]$ and $\ell \in [t]$, follow this procedure:

> *Partition $[t]$ into subsets $E, E_1, E_2, \ldots, E_m$ so that the size of E_j is precisely $e(j)$, and $E = [t] \setminus \bigcup_{j\in[m]} E_j$. For each $j \in [m]$ and each $\ell \in E_j$, set $\eta_j[e_\ell] = 1$, and for $j' \neq j$, set $\eta_{j'}[e^\ell] = 0$.*

Remark 2.5. *Let $m \in \mathbb{Z}^+$. For an arbitrary multiset $\{\eta_j\}_{j=1}^m$ of binary strings built using Definition 2.4, for each $\ell \in [t]$, there is a unique $j \in [m]$ such that $\eta_j[e_\ell] = 1$. Consequently, each $\mu \in \mathcal{M}(\{\eta_j\}_{j=1}^m)$ will have $\mu[e_\ell] = 0$ for all $\ell \in [t]$ because μ must minimize $\sum_{j\in[m]} H(\eta_j, \mu)$.*

Definition 2.6. *Let $n \in \mathbb{Z}^+$ and $t \in \mathbb{Z}^{\geq 0}$ be arbitrary. Two binary strings η and $\overline{\eta}$ with coordinates $(x_1, y_1, \ldots, x_n, y_n, e_1, \ldots, e_t)$, are said to be* complementary on the first $2n$ coordinates *if $\eta[x_i] = 1 - \overline{\eta}[x_i]$ and $\eta[y_i] = 1 - \overline{\eta}[y_i]$ for each $i \in [n]$.*

The following fact will be useful.

Fact 2.7. *Let η and $\overline{\eta}$ be binary strings on coordinates*

$$(x_1, y_1, \ldots, x_n, y_n, e_1, \ldots, e_t).$$

Set $e(\eta) := \sum_{i\in[t]} \eta[e_i]$, the number of additional ones in η. Define $e(\overline{\eta})$ similarly. If η and $\overline{\eta}$ are complementary on the first $2n$ coordinates, then for any $\mu \in \{0,1\}^{2n} \times \{0\}^t$,

$$H(\mu,\eta) + H(\mu,\overline{\eta}) = 2n + e(\eta) + e(\overline{\eta}).$$

Proof. For each $i \in [n]$, either $\mu[x_i] = \eta[x_i]$ or $\mu[x_i] = \overline{\eta}[x_i]$, but not both. This is also true for each y_i. This accounts for the $2n$ in the sum. Because $\mu[e_i] = 0$ for all $i \in [t]$, each $i \in [t]$ with $\eta[e_i] = 1$ will contribute one to the sum. Also each $i \in [t]$ with $\overline{\eta}[e_i]$ will contribute one to the sum. This completes the proof. ∎

Definition 2.8. *Given an arbitrary multiset of binary strings S, we say that a coordinate s is* ambiguous *if there are exactly $\frac{1}{2}|S|$ binary strings $\eta \in S$, counted with multiplicity, such that $\eta[s] = 0$. Consequently, if you change the value of a median at an ambiguous coordinate, you obtain another median.*

Fact 2.9. *Let S be a multiset of binary strings which are defined on the coordinates*

$$(x_1, y_1, \ldots, x_n, y_n, e_1, \ldots, e_t).$$

If S can be partitioned into pairs of vertices where the two strings in a pair are complementary on the first $2n$ coordinates, then each x_i and each y_i is an ambiguous coordinate.

Fix an arbitrary D3CNF, Γ, with n variables and k clauses. Fix a clause c_i in Γ. For this clause, we are now ready to define a set of 50 strings

$$\mathcal{C}_i = \{\nu_1^i, \nu_2^i, \ldots, \nu_{50}^i\}.$$

First assume that $c_i = v_\alpha \vee v_\beta \vee v_\gamma$, a disjunction of three positive literals. Because Γ is a D3CNF, we may assume $\alpha < \beta < \gamma$.

For each $j \in [50]$, we will supply the following three pieces of information for ν_j^i:

(a) The values for $\nu_j^i[x_\alpha], \nu_j^i[y_\alpha], \nu_j^i[x_\beta], \nu_j^i[y_\beta], \nu_j^i[x_\gamma], \nu_j^i[y_\gamma]$ will be explicitly defined.

(b) A constant $\kappa_{ij} \in \{0,1\}$ will be given so that $\nu_j^i[x_\ell] = \nu_j^i[y_{\ell'}] = \kappa_{ij}$ for all $\ell, \ell' \in [n] \setminus \{\alpha, \beta, \gamma\}$.

(c) The string will be assigned some number of additional ones.

By Definition 2.4, this is sufficient to explicitly define ν_j^i.

The Table 2.1, there is a row for each string in $\mathcal{C}_i$. The three defining pieces of information are found in Columns (A), (B), and (C) of Table 2.1. The remainder of the table will be explained in Subsection 2.1.2.

For each $j \in [50]$, row j of Table 2.1 supplies the three ingredients needed to define ν_j^i. By matching the 6-bit string in Column A of row j with

$$(\nu_j^i[x_\alpha], \nu_j^i[y_\alpha], \nu_j^i[x_\beta], \nu_j^i[y_\beta], \nu_j^i[x_\gamma], \nu_j^i[y_\gamma])$$

we obtain the 6 values for (a). The constant κ_{ij} for (b) is found in Column (B) of row j. For (c), the number of additional ones in ν_j^i is found in Column C of row j.

With a slight modification in the reading of Column A, the 50 rows of Table 2.1 will also supply the 50 strings for a clause which contains negative literals. Fix an arbitrary clause c_i in Γ which now may have negative literals. For each $j \in [50]$, the definition of string ν_j^i will again be based on Columns A, B, C of row j in Table 2.1. The same information will be gleaned from Columns B and C as in the case when c_i had no negative literals. The only difference is with Column A which will be explained next.

Let S_i denote the *support set* of clause c_i. If c_i contains the variables $v_\alpha, v_\beta, v_\gamma$, then $S_i := \{x_\alpha, y_\alpha, x_\beta, y_\beta, x_\gamma, y_\gamma\}$. Clause c_i must be one of the 8 clauses listed in Column A of Table 2.2. For $j \in [50]$, ν_j^i is defined on the coordinates S_i by matching the entry in the right column of the c_i row of Table 2.2 with the 6-bit string in Column A of the j^{th} row of Table 2.1.

Table 2.1 The 50 strings in $\mathcal{C}_i$ for a single clause c_i along with their Hamming distance from medians in $\mathcal{M}'$.

		A	**B**	**C**	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**	**M8**
	Row #	Values of ν_j^i on its support set	$\nu_j^i[x_\ell]$, $\nu_j^i[y_\ell]$ $(v_\ell \notin c_i)$	Add'l Ones	10 10 10	10 10 01	10 01 10	01 10 10	10 01 01	01 10 01	01 01 10	01 01 01
$\mathcal{N}_1^{(+3)}$	1	01 00 00	0	+3	$n+4$	$n+4$	$n+4$	$n+2$	$n+4$	$n+2$	$n+2$	$n+2$
	2	00 01 00	0	+3	$n+4$	$n+4$	$n+2$	$n+4$	$n+2$	$n+4$	$n+2$	$n+2$
	3	00 00 01	0	+3	$n+4$	$n+2$	$n+4$	$n+4$	$n+2$	$n+2$	$n+4$	$n+2$
$\overline{\mathcal{N}_1^{(+0)}}$	4	10 11 11	1	+0	$n-1$	$n-1$	$n-1$	$n+1$	$n-1$	$n+1$	$n+1$	$n+1$
	5	11 10 11	1	+0	$n-1$	$n-1$	$n+1$	$n-1$	$n+1$	$n-1$	$n+1$	$n+1$
	6	11 11 10	1	+0	$n-1$	$n+1$	$n-1$	$n-1$	$n+1$	$n+1$	$n-1$	$n+1$
$\mathcal{I}_2^{(+2)} \setminus \mathcal{N}_2^{(+2)}$	7	10 10 00	0	+2	n	n	$n+2$	$n+2$	$n+2$	$n+2$	$n+4$	$n+4$
	8	10 00 10	0	+2	n	$n+2$	n	$n+2$	$n+2$	$n+4$	$n+2$	$n+4$
	9	00 10 10	0	+2	n	$n+2$	$n+2$	n	$n+4$	$n+2$	$n+2$	$n+4$
	10	10 10 00	0	+2	n	n	$n+2$	$n+2$	$n+2$	$n+2$	$n+4$	$n+4$
	11	10 00 01	0	+2	$n+2$	n	$n+2$	$n+4$	n	$n+2$	$n+4$	$n+2$
	12	00 10 01	0	+2	$n+2$	n	$n+4$	$n+2$	$n+2$	n	$n+4$	$n+2$
	13	10 01 00	0	+2	$n+2$	$n+2$	n	$n+4$	n	$n+4$	$n+2$	$n+2$
	14	10 00 10	0	+2	n	$n+2$	n	$n+2$	$n+2$	$n+4$	$n+2$	$n+4$
	15	00 01 10	0	+2	5	$n+4$	n	$n+2$	$n+2$	$n+4$	n	$n+2$
	16	01 10 00	0	+2	$n+2$	$n+2$	$n+4$	n	$n+4$	n	$n+2$	$n+2$
	17	01 00 10	0	+2	$n+2$	$n+4$	$n+2$	n	$n+4$	$n+2$	n	$n+2$
	18	00 10 10	0	+2	n	$n+2$	$n+2$	n	$n+4$	$n+2$	$n+2$	$n+4$
	19	10 01 00	0	+2	$n+2$	$n+2$	n	$n+4$	n	$n+4$	$n+2$	$n+2$
	20	10 00 01	0	+2	$n+2$	n	$n+2$	$n+4$	n	$n+2$	$n+4$	$n+2$
	21	00 01 01	0	+2	$n+4$	$n+4$	$n+2$	$n+4$	n	$n+2$	$n+2$	n
	22	01 10 00	0	+2	$n+2$	$n+2$	$n+4$	n	$n+4$	n	$n+2$	$n+2$
	23	01 00 01	0	+2	$n+4$	$n+2$	$n+4$	$n+2$	$n+2$	n	$n+2$	n
	24	00 10 01	0	+2	$n+2$	n	$n+4$	$n+2$	$n+2$	n	$n+4$	$n+2$
	25	01 01 00	0	+2	$n+4$	$n+4$	$n+2$	$n+2$	$n+2$	$n+2$	n	n
	26	01 00 10	0	+2	$n+2$	$n+4$	$n+2$	n	$n+4$	$n+2$	n	$n+2$
	27	00 01 10	0	+2	$n+2$	$n+4$	n	$n+2$	$n+2$	$n+4$	n	$n+2$
$\overline{\mathcal{I}_2^{(+1)}} \setminus \overline{\mathcal{N}_2^{(+1)}}$	28	10 10 11	1	+1	$n-1$	$n-1$	$n+1$	$n+1$	$n+1$	$n+1$	$n+3$	$n+3$
	29	10 11 01	1	+1	$n+1$	$n-1$	$n+1$	$n+3$	$n-1$	$n+1$	$n+3$	$n+1$
	30	11 10 01	1	+1	$n+1$	$n-1$	$n+3$	$n+1$	$n+1$	$n-1$	$n+3$	$n+1$
	31	10 01 11	1	+1	$n+1$	$n+1$	$n-1$	$n+3$	$n-1$	$n+3$	$n+1$	$n+1$
	32	10 11 10	1	+1	$n-1$	$n+1$	$n-1$	$n+1$	$n+1$	$n+3$	$n+1$	$n+3$
	33	11 01 10	1	+1	$n+1$	$n+3$	$n-1$	$n+1$	$n+1$	$n+3$	$n-1$	$n+1$
	34	01 10 11	1	+1	$n+1$	$n+1$	$n+3$	$n-1$	$n+3$	$n-1$	$n+1$	$n+1$
	35	01 11 10	1	+1	$n+1$	$n+3$	$n+1$	$n-1$	$n+3$	$n+1$	$n-1$	$n+1$
	36	11 10 10	1	+1	$n-1$	$n+1$	$n+1$	$n-1$	$n+3$	$n+1$	$n+1$	$n+3$
	37	10 01 11	1	+1	$n+1$	$n+1$	$n-1$	$n+3$	$n-1$	$n+3$	$n+1$	$n+1$
	38	10 11 01	1	+1	$n+1$	$n-1$	$n+1$	$n+3$	$n-1$	$n+1$	$n+3$	$n+1$
	39	11 01 01	1	+1	$n+3$	$n+1$	$n+1$	$n+3$	$n-1$	$n+1$	$n+1$	$n-1$
	40	01 10 11	1	+1	$n+1$	$n+1$	$n+3$	$n-1$	$n+3$	$n-1$	$n+1$	$n+1$
	41	01 11 01	1	+1	$n+3$	$n+1$	$n+3$	$n+1$	$n+1$	$n-1$	$n+1$	$n-1$
	42	11 10 01	1	+1	$n+1$	$n-1$	$n+3$	$n+1$	$n+1$	$n-1$	$n+3$	$n+1$
	43	01 01 11	1	+1	$n+3$	$n+3$	$n+1$	$n+1$	$n+1$	$n+1$	$n-1$	$n-1$
	44	01 11 10	1	+1	$n+1$	$n+3$	$n+1$	$n-1$	$n+3$	$n+1$	$n-1$	$n+1$
	45	11 01 10	1	+1	$n+1$	$n+3$	$n-1$	$n+1$	$n+1$	$n+3$	$n-1$	$n+1$
	46	01 01 11	1	+1	$n+3$	$n+3$	$n+1$	$n+1$	$n+1$	$n+1$	$n-1$	$n-1$
	47	01 11 01	1	+1	$n+3$	$n+1$	$n+3$	$n+1$	$n+1$	$n-1$	$n+1$	$n-1$
	48	11 01 01	1	+1	$n+3$	$n+1$	$n+1$	$n+3$	$n-1$	$n+1$	$n+1$	$n-1$
$\mathcal{N}_3^{(+1)}$	49	01 01 01	0	+1	$n+3$	$n+2$	$n+2$	$n+4$	n	n	n	$n-2$
$\overline{\mathcal{N}_3^{(+2)}}$	50	10 10 10	1	+2	$n-1$	$n+1$	$n+1$	$n+1$	$n+3$	$n+3$	$n+3$	$n-3$

For a clause c_i, the left three columns define the 50 strings in $\mathcal{C}_i$. In row j, the 6-bit string gives the values of ν_j^i on the support set S_i as described by Table 2.2. The second column gives the constant value to be assigned to all x_ℓ and y_ℓ which are not in S_i. The third column specifies the number of extra ones in ν_j^i. The collection $\{01, 10\}^3$ is listed along the top row. The entry in row j and column ℓ is the number of additional ones in ν_j^i added to the Hamming distance between the 6-bit string in row j and the 6-bit string at the top of column ℓ.

Example 2.10. *For an example, when $c_i = v_\alpha \vee \overline{v_\beta} \vee \overline{v_\gamma}$, the last row of Table 2.1 says that the string ν^i_{50} must have*

$$(\nu^i_{50}[x_\alpha], \nu^i_{50}[y_\alpha], \nu^i_{50}[y_\beta], \nu^i_{50}[x_\beta], \nu^i_{50}[y_\gamma], \nu^i_{50}[x_\gamma]) = (101010).$$

Therefore, $\nu^i_{50}[x_\alpha] = 1$, $\nu^i_{50}[y_\alpha] = 0$, $\nu^i_{50}[x_\beta] = 0$, $\nu^i_{50}[y_\beta] = 1$, $\nu^i_{50}[x_\gamma] = 0$, and $\nu^i_{50}[y_\gamma] = 1$. Further, Column B implies $\nu^i_{50}(x_\ell) = \nu^i_{50}(y_\ell) = 1$ for all $\ell \in [n] \setminus \{\alpha, \beta, \gamma\}$ and, from Column C, ν^i_{50} will have 2 additional ones.

Now that we have defined $\mathcal{C}_i$ for any clause c_i, let us analyze $\mathcal{M}(\mathcal{C}_i)$. By Fact 2.5, for every $\mu \in \mathcal{M}(\mathcal{C}_i)$ and $\ell \in [t]$, $\mu[e_\ell] = 0$.

In Column B of Table 2.1, it is evident that for any $\ell \in [n] \setminus \{\alpha, \beta, \gamma\}$, the number of strings ν^i_j with $\nu^i_j[x_\ell] = 0$ is $25 = \frac{1}{2}|\mathcal{C}_i|$. Therefore, by Definition 2.8, the coordinates x_ℓ and y_ℓ are ambiguous. Through careful inspection of the strings in Column A of Table 2.1, we see that coordinates x'_ℓ and y'_ℓ are also ambiguous for each $\ell' \in \{\alpha, \beta, \gamma\}$. Therefore we have proven the following fact, which was one of our goals:

Fact 2.11. *For an arbitrary clause c_i with three distinct variables,*

$$\mathcal{M}(\mathcal{C}_i) = \{0,1\}^{2n} \times \{0\}^t.$$

Remark 2.12. *By visual inspection of Table 2.1, the binary strings $\mathcal{C}_i$ can be partitioned into pairs where the two strings in a pair are complementary on the first $2n$ coordinates.*

2.1.2 Hamming distances between C_i and possible medians

Here we explain the remainder of Table 2.1. Fix a clause c_i in Γ which will be used throughout this subsection. Suppose c_i has variables v_α, v_β, and v_γ. By Fact 2.11,

Table 2.2 A key for interpreting Column A of Table 2.1.

Clause	Key to interpret Column A of Table 2.1
$v_\alpha \vee v_\beta \vee v_\gamma$	$(\nu_j^i[x_\alpha], \nu_j^i[y_\alpha], \nu_j^i[x_\beta], \nu_j^i[y_\beta], \nu_j^i[x_\gamma], \nu_j^i[y_\gamma])$
$\overline{v_\alpha} \vee v_\beta \vee v_\gamma$	$(\nu_j^i[y_\alpha], \nu_j^i[x_\alpha], \nu_j^i[x_\beta], \nu_j^i[y_\beta], \nu_j^i[x_\gamma], \nu_j^i[y_\gamma])$
$v_\alpha \vee \overline{v_\beta} \vee v_\gamma$	$(\nu_j^i[x_\alpha], \nu_j^i[y_\alpha], \nu_j^i[y_\beta], \nu_j^i[x_\beta], \nu_j^i[x_\gamma], \nu_j^i[y_\gamma])$
$v_\alpha \vee v_\beta \vee \overline{v_\gamma}$	$(\nu_j^i[x_\alpha], \nu_j^i[y_\alpha], \nu_j^i[x_\beta], \nu_j^i[y_\beta], \nu_j^i[y_\gamma], \nu_j^i[x_\gamma])$
$\overline{v_\alpha} \vee \overline{v_\beta} \vee v_\gamma$	$(\nu_j^i[y_\alpha], \nu_j^i[x_\alpha], \nu_j^i[y_\beta], \nu_j^i[x_\beta], \nu_j^i[x_\gamma], \nu_j^i[y_\gamma])$
$\overline{v_\alpha} \vee v_\beta \vee \overline{v_\gamma}$	$(\nu_j^i[y_\alpha], \nu_j^i[x_\alpha], \nu_j^i[x_\beta], \nu_j^i[y_\beta], \nu_j^i[y_\gamma], \nu_j^i[x_\gamma])$
$v_\alpha \vee \overline{v_\beta} \vee \overline{v_\gamma}$	$(\nu_j^i[x_\alpha], \nu_j^i[y_\alpha], \nu_j^i[y_\beta], \nu_j^i[x_\beta], \nu_j^i[y_\gamma], \nu_j^i[x_\gamma])$
$\overline{v_\alpha} \vee \overline{v_\beta} \vee \overline{v_\gamma}$	$(\nu_j^i[y_\alpha], \nu_j^i[x_\alpha], \nu_j^i[y_\beta], \nu_j^i[x_\beta], \nu_j^i[y_\gamma], \nu_j^i[x_\gamma])$

For any clause in the left column, the corresponding entry in the right column above will be matched with the 6-bit string in Column A of row j of Table 2.1 to determine the value of ν_j^i at each bit in the support set S_i.

$\mathcal{M}(\mathcal{C}_i) = \{0,1\}^{2n} \times \{0\}^t$. Therefore, $\mathcal{M}'(\mathcal{C}_i)$, from Definition 1.9, must be equal to $\{01,10\}^n \times \{0\}^t$. For this subsection, define

$$\mathcal{M} := \mathcal{M}(\mathcal{C}_i), \qquad \mathcal{M}' := \mathcal{M}'(\mathcal{C}_i).$$

Define an equivalence relation $\sim_i$ on $\mathcal{M}'$ such that two medians are equivalent if they agree on the coordinates in the support set S_i of c_i. The result will be 8 equivalence classes because $\mu[x_\ell] \neq \mu[y_\ell]$ for each $\ell \in \{\alpha, \beta, \gamma\}$ for each $\mu \in \mathcal{M}'$.

Here we define a one-to-one correspondence between the equivalence classes of $\mathcal{M}'$ under $\sim_i$ and the 6-bit strings heading Columns M1 through M8 in Table 2.1.

Definition 2.13. *Fix a clause c_i and an integer $\ell \in [8]$. Consider the 6-bit string δ which heads column Mℓ. In Table 2.2, locate the tuple in the right column corresponding to our fixed clause c_i. After replacing each ν_j^i with μ in the tuple, match this tuple with δ. This gives six values that a median $\mu \in \mathcal{M}'$ must have if it is in the equivalence class represented by the column heading δ.*

In Definition 2.1, we defined a correspondence between $\mathcal{M}'$ and truth assignments for Γ. In Definition 2.3, we introduced the notation $\mathcal{M}'_\Gamma(\mathcal{C}_i)$ for the collection of medians in $\mathcal{M}'$ which correspond to satisfying truth assignments for Γ. Similarly, we

defined $\mathcal{M}'_{c_i}(\mathcal{C}_i)$ for each clause c_i in Γ. For the remainder of this subsection, set

$$\mathcal{M}'_\Gamma := \mathcal{M}'_\Gamma(\mathcal{C}_i), \qquad \mathcal{M}'_{c_i} := \mathcal{M}'_{c_i}(\mathcal{C}_i).$$

The following claim uses the correspondence in Definition 2.13 to connect $\mathcal{M}' \setminus \mathcal{M}'_{c_i}$ with a particular equivalence class.

Claim 2.14. *Let c_i be a clause in Γ. For any $\mu \in \mathcal{M}'$, μ is in the equivalence class represented by Column M8 of Table 2.1 if and only if $\mu \in \mathcal{M}' \setminus \mathcal{M}'_{c_i}$.*

Proof. Fix a clause c_i with variables $v_\alpha, v_\beta, v_\gamma$. This clause may have some negative literals. We focus our attention on v_α. The arguments for v_β and v_γ are exactly the same.

There are two cases depending on whether v_α appears as a positive literal or a negative literal in c_i.

In the case where v_α appears in c_i as a positive literal, the truth assignment which makes c_i false assigns a value of false to v_α. A corresponding median $\mu \in \mathcal{M}'$ has $\mu[x_\alpha] = 0$ and $\mu[y_\alpha] = 1$. Because v_α appears as a positive literal in c_i, the entry in the second column of Table 2.2 has $\mu[x_\alpha]$ followed by $\mu[y_\alpha]$. So, in this case, the 6-bit string which heads the column for medians in $\mathcal{M}' \setminus \mathcal{M}'_{c_i}$ has 01 in the first two entries.

In the case where v_α appears as a negative literal in c_i, the non-satisfying truth assignments for c_i must have v_α true. The corresponding medians $\mu \in \mathcal{M}'$ will have $\mu[x_\alpha] = 1$ and $\mu[y_\alpha] = 0$. For the clauses with variable v_α appearing as a negative literal in c_i, a quick glance at Table 2.2 reveals that $\mu[y_\alpha]$ immediately precedes $\mu[x_\alpha]$ in the 6-bit column headings in Table 2.1. As a result, the column representing medians in $\mathcal{M}' \setminus \mathcal{M}'_{c_i}$ has 01 in the first two entries.

Repeating this argument for v_β and v_γ, we see that medians in $\mathcal{M}' \setminus \mathcal{M}'_{c_i}$ are represented by the column with heading 010101. ∎

Now that we have defined the rows and columns of Table 2.1, we conclude this subsection by defining the entries within Table 2.1 for fixed clause c_i.

Let $\mu \in \mathcal{M}'$ be an arbitrary median that falls into the equivalence class represented by Column Mℓ for some $\ell \in [8]$. The entry $a_{j\ell}$ in Row j and Column Mℓ of Table 2.1 is $H(\mu, \nu_j^i)$. This value can be calculated as follows:

- First, take the Hamming distance between the 6-bit string in Column A of Row j and the 6-bit string in the header of Column Mℓ. This is equal to the Hamming distance between the restrictions of μ and ν_j^i to the support set S_i for c_i.

- For any $s \notin \{\alpha, \beta, \gamma\}$, $\mu[x_s] \neq \mu[y_s]$ and $\nu_j^i[x_s] = \nu_j^i[y_s]$. Therefore the Hamming distance between $(\mu[x_s], \mu[y_s])$ and $(\nu_j^i[x_s], \nu_j^i[y_s])$ is 1 for each $s \in [n] \setminus \{\alpha, \beta, \gamma\}$.

- Finally, because $\mu[e_s] = 0$ for all $s \in [t]$, the Hamming distance between the restrictions of μ and ν_j^i to the coordinates $(e_1, e_2, \ldots, e_t)$ is the number of additional ones in ν_j^i which is found in Column C of Row j.

Adding these three values together gives the entry $a_{j\ell}$.

2.1.3 Distinguishing the satisfying truth assignments

Fix a clause c_i in arbitrary D3CNF Γ. For this subsection, we again set $\mathcal{M}' := \mathcal{M}'(\mathcal{C}_i)$, $\mathcal{M}'_\Gamma := \mathcal{M}'_\Gamma(\mathcal{C}_i)$, and $\mathcal{M}'_{c_i} := \mathcal{M}'_{c_i}(\mathcal{C}_i)$. For each $\mu \in \mathcal{M}' \setminus \mathcal{M}'_{c_i}$, μ is in the equivalence class represented by 010101 according to Claim 2.14. Then reading the entries in Column M8 of Table 2.1, we find

$$\{H(\mu, \nu_j^i) : j \in [50]\} = \{(n-2)_{(1)}, (n-1)_{(6)}, n_{(3)}, (n+1)_{(15)}, \\ (n+2)_{(15)}, (n+3)_{(3)}, (n+4)_{(6)}, (n+5)_{(1)}\}. \tag{2.1}$$

Otherwise, for each median $\mu \in \mathcal{M}'_{c_i}$, μ is in one of 7 equivalence classes represented in Columns M1 through M7. The entries in each of these columns yields

$$\{H(\mu, \nu^i_j) : j \in [50]\} = \{(n-1)_{(7)}, n_{(6)}, (n+1)_{(12)}, (n+2)_{(12)}, (n+3)_{(6)}, (n+4)_{(7)}\}. \tag{2.2}$$

Therefore,we can use $\mathcal{C}_i$ to distinguish between the medians in $\mathcal{M}'_{c_i}$ and the medians in $\mathcal{M}' \setminus \mathcal{M}'_{c_i}$. For example, given $\mu \in \mathcal{M}' = \{01, 10\}^n \times \{0\}^t$, if we determine that $(n+5) \in \{H(\mu, \nu^i_j) : j \in [50]\}$, then we can conclude $\mu \in \mathcal{M}' \setminus \mathcal{M}'_{c_i}$.

Now we wish to consider all of the C_i multisets together. It is clear that each x_i and each y_i coordinates will remain ambiguous in the multiset $\uplus_{i\in[k]} \mathcal{C}_i$. For the additional ones, we will take t large enough to maintain the property that, for each $i \in [t]$, there is at most one binary string η in $\uplus_{i\in[k]} \mathcal{C}_i$ with $\eta[e_i] = 1$. As a result,

$$\mathcal{M}\left(\biguplus_{i\in[k]} \mathcal{C}_i\right) = \{0,1\}^n \times \{0\}^t.$$

Further,

$$\mathcal{M}'_{c_i} := \mathcal{M}'_{c_i}(\mathcal{C}_i) = \mathcal{M}'_{c_i}\left(\biguplus_{i\in[k]} \mathcal{C}_i\right),$$
$$\mathcal{M}'_{\Gamma} := \mathcal{M}'_{\Gamma}(\mathcal{C}_i) = \mathcal{M}'_{\Gamma}\left(\biguplus_{i\in[k]} \mathcal{C}_i\right).$$

By definition of the sets $\mathcal{M}'_{c_i}$ and $\mathcal{M}'_{\Gamma}$,

$$\mathcal{M}'_{\Gamma} = \bigcap_{i\in[k]} \mathcal{M}'_{c_i}, \tag{2.3}$$

$$\mathcal{M}' \setminus \mathcal{M}'_{\Gamma} = \mathcal{M}' \setminus \bigcap_{i\in[k]} \mathcal{M}'_{c_i} = \bigcup_{i\in[k]} \mathcal{M}' \setminus \mathcal{M}'_{c_i}. \tag{2.4}$$

Therefore the multiset $\uplus_{i\in[k]} \mathcal{C}_i$ will serve as a tool to distinguish $\mathcal{M}'_{\Gamma}$ from $\mathcal{M}' \setminus \mathcal{M}'_{\Gamma}$.

2.2 Complexity result for #StarSPSCJ

Before stating Theorem 2.18, we need a result which is equivalent to the Prime Number Theorem. Define

$$\theta(x) := \sum_{\substack{p \leq x \\ p \text{ prime}}} \log p.$$

Theorem 2.15. $\theta(x) \sim x$.

As a result, the next lemma and corollary hold.

Lemma 2.16 (Rosser (1941))**.** *For* $2 \leq x$*,*

$$\left(1 - \frac{2.85}{\log x}\right) x \leq \theta(x) \leq \left(1 + \frac{2.85}{\log x}\right) x.$$

Corollary 2.17. *For any* $n \geq 300$*,*

$$e^{n/2} \leq \prod_{\substack{p \leq n \\ p \text{ prime}}} p \leq e^{3n/2}.$$

Now we can prove the main result for this chapter.

Theorem 2.18. *#StarSPSCJ is #P-complete.*

Proof. We have already verified that #StarSPSCJ, which is a subproblem of #SPSCJ, is in #P in Lemma 1.23. To show #P-complete, we give a polynomial time reduction from #D3SAT. Fix an arbitrary D3CNF $\Gamma = c_1 \wedge c_2 \wedge \ldots \wedge c_k$ where each c_i is a clause and Γ has n variables.

Using the bound in Corollary 2.17, let $n' = \max\{300, n+5\}$. Fix a prime number p which is greater than n' and at most $5n'$. Let

$$q := p - (n + 5).$$

We will explicitly define a multiset $\mathcal{D}(p) = \mathcal{A}(p) \cup \bigcup_{i\in[n]} \mathcal{B}_i(p) \cup \bigcup_{i\in[k]} \mathcal{C}_i(p)$ consisting of $2 + 2n + 50k$ binary strings with coordinates

$$(x_1, y_1, x_2, y_2, \ldots, x_n, y_n, e_1, \ldots, e_{t(p)})$$

where

$$t(p) := 2(q + 4) + 2n(q + 3) + k(75 + 50q). \tag{2.5}$$

The coordinates $e_1, e_2, \ldots, e_{t(p)}$ are for the *additional ones*. In order to define each $\eta \in \mathcal{D}(p)$, we will give exact values for $\eta[x_j]$ and $\eta[y_j]$ for each $j \in [n]$ and specify the number of additional ones that η will have. Definition 2.4 tells how to obtain the values of $\eta[e_j]$ for each $j \in [t(p)]$ from this information.

All strings in $\mathcal{D}(p)$ will come in pairs which are complementary on the first $2n$ entries (Definition 2.8). As a result, we can use Fact 2.9 to see that each of the first $2n$ coordinates are ambiguous in $\mathcal{D}(p)$.

Now we begin defining the strings in multiset that together create $\mathcal{D}(p)$. The set $\mathcal{A}(p)$ consists of two strings, α and $\overline{\alpha}$. Define α to have $\alpha[x_i] = \alpha[y_i] = 1$ for all $i \in [n]$ and $q + 4$ additional ones. Define $\overline{\alpha}$ to be complementary to α on the first $2n$ entries and have $q + 4$ additional ones.

For each $j \in [n]$, the set $\mathcal{B}_j(p)$ will consist of two strings, β_j and $\overline{\beta_j}$. Define β_j to be the string with $\beta_j[x_j] = \beta_j[y_j] = 1$ and for all $j' \in [n]$ with $j' \neq j$, $\beta_j[x_{j'}] = \beta_j[y_{j'}] = 0$ and $q+3$ additional ones. Define $\overline{\beta_j}$ to be complementary to β_j on the first $2n$ entries and have $q + 3$ additional ones.

For each $i \in [k]$, the set $\mathcal{C}_i(p)$ will have 50 strings. These are obtained by adding q more additional ones to the 50 strings in $\mathcal{C}_i$ which were defined through Table 2.1 (see Section 2.1.1). In other words, increase each entry in Column C of Table 2.1 by q to obtain $\mathcal{C}_i(p)$.

In summary, we have constructed the strings

$$\mathcal{D}(p) := \mathcal{A}(p) \cup \bigcup_{i\in[n]} \mathcal{B}_i(p) \cup \bigcup_{i\in[k]} \mathcal{C}_i(p).$$

As described in Definitions 1.10 and 2.3 and for each clause c_i in Γ, set

$$\mathcal{M}(p) := \mathcal{M}(\mathcal{D}(p)), \qquad \mathcal{M}'(p) := \mathcal{M}'(\mathcal{D}(p)),$$
$$\mathcal{M}'_{c_i}(p) := \mathcal{M}'_{c_i}(\mathcal{D}(p)), \qquad \mathcal{M}'_{\Gamma}(p) := \mathcal{M}'_{\Gamma}(\mathcal{D}(p)).$$

As stated in Fact 2.5, each $\mu \in \mathcal{M}(p)$ has $\mu[e_j] = 0$ for all $j \in [t(p)]$. Additionally, because all of the strings in $\mathcal{D}(p)$ come in complementary pairs, the coordinates x_j and y_j are ambiguous for each $j \in [n]$ (Fact 2.9). Thus there are 2^{2n} medians μ. More precisely,

$$\mathcal{M}(p) = \{0,1\}^{2n} \times \{0\}^{t(p)} \text{ and} \tag{2.6}$$
$$\mathcal{M}'(p) = \{01,10\}^{n} \times \{0\}^{t(p)}.$$

Define

$$\mathcal{H}(\mu, \mathcal{A}(p)) := \prod_{a \in \mathcal{A}(p)} H(\mu, a)!$$

and likewise define $\mathcal{H}(\mu, \mathcal{B}_j(p))$ and $\mathcal{H}(\mu, \mathcal{C}_i(p))$ for each $j \in [n]$ and $i \in [k]$. Therefore the number of SCJ scenarios admitted by μ (Definition 1.8) can be expressed by

$$\mathcal{H}(\mu) := \mathcal{H}(\mu, \mathcal{A}(p)) \cdot \prod_{i \in [n]} \mathcal{H}(\mu, \mathcal{B}_i(p)) \cdot \prod_{i \in [k]} \mathcal{H}(\mu, \mathcal{C}_i(p)).$$

At this point, we wish to calculate $d(\mu) \mod p$ for each median $\mu \in \mathcal{M}(p)$. To analyze $\mathcal{H}(\mu)$ for each $\mu \in \mathcal{M}$, we define the following 3 properties that a median $\mu \in \mathcal{M}(p)$ may have.

Property 1. $\sum_{i \in [n]} (\mu[x_i] + \mu[y_i]) = n$.

Property 2. $\mu \in \mathcal{M}'(p)$.

Property 3. $\mu \in \mathcal{M}'_{\Gamma}(p)$.

First notice that these properties are nested. Any $\mu \in \mathcal{M}(p)$ with Property 2 must also have Property 1. Likewise, if μ has Property 3, it will also have Property 2. The next 4 claims divide $\mathcal{M}(p)$ into 4 classes and examine $\mathcal{H}(\mu)$ for medians in each class.

Claim 2.19. *For arbitrary $\mu \in \mathcal{M}(p)$, if μ does not have Property 1, and consequently does not have Property 2 or 3, then $\mathcal{H}(\mu) \equiv 0 \mod p$.*

Proof. Let μ be an arbitrary median in $\mathcal{M}(p)$. For $\alpha \in \mathcal{A}(p)$, Fact 2.7 gives

$$H(\mu, \alpha) + H(\mu, \overline{\alpha}) = 2n + (q+4) + (q+4) = 2p - 2.$$

Hence, there is an integer r such that $q + 4 \leq r \leq 2n + q + 4$ and $H(\mu, \alpha) = r$ with

$$\mathcal{H}(\mu, \mathcal{A}(p)) = r!(2p - 2 - r)!.$$

Since μ does not have Property 1, we can conclude that exactly one of the following holds:

$$H(\mu, \alpha) \geq (n+1) + (q+4) = p$$
$$H(\mu, \overline{\alpha}) \geq (n+1) + (q+4) = p.$$

Therefore, either $r \geq p$ or $(2p - 2 - r) \geq p$. In the first case, $r!$ is divisible by p and, in the second, $(2p - 2 - r)!$ is divisible by p. Therefore $\mathcal{H}(\mu, \mathcal{A}(p)) \equiv 0 \mod p$ and consequently $\mathcal{H}(\mu) \equiv 0 \mod p$. ∎

Claim 2.20. *For an arbitrary $\mu \in \mathcal{M}(p)$, if μ has Property 1, but does not have Property 2, then $\mathcal{H}(\mu) \equiv 0 \mod p$.*

Proof. Suppose $\mu \in \mathcal{M}(p) \setminus \mathcal{M}'(p)$ but μ has Property 1. Because $\mu \notin \mathcal{M}'(p)$, there is an integer $j_0 \in [n]$ such that $\mu[x_{j_0}] = \mu[y_{j_0}]$. In the case when $\mu[x_{j_0}] = 0$, we have $H(\mu, \beta_{j_0}) = (n+2) + (q+3) = p$. Otherwise $\mu[x_i] = 1$ which implies $H(\mu, \overline{\beta_{j_0}}) = (n+2) + (q+3) = p$. In either case,

$$\mathcal{H}(\mu, \mathcal{B}_{j_0}) = p!(p-4)!$$

and consequently $\mathcal{H}(\mu) \equiv 0 \mod p$. ∎

Claim 2.21. *For an arbitrary $\mu \in \mathcal{M}(p)$, if μ has Properties 1 and 2, but does not have Property 3, then $\mathcal{H}(\mu) \equiv 0 \mod p$.*

Proof. Let μ be in $\mathcal{M}'(p) \setminus \mathcal{M}'_{\Gamma}(p)$. Since μ corresponds to a truth assignment which does not satisfy Γ, there is a clause c_{i_0} in Γ which is not satisfied by this truth assignment. Therefore $\mu \in \mathcal{M}'(p) \setminus \mathcal{M}'_{c_{i_0}}(p)$. By (2.1), before adding the q additional ones to each string from $\mathcal{C}_{i_0}$, we have

$$\{H(\mu, \nu_j^{i_0}) : \nu_j^{i_0} \in \mathcal{C}_{i_0}\} = \{(n-2)_{(1)}, (n-1)_{(6)}, n_{(3)}, (n+1)_{(15)}, \\ (n+2)_{(15)}, (n+3)_{(3)}, (n+4)_{(6)}, (n+5)_{(1)}\}. \tag{2.7}$$

To create $\mathcal{C}_{i_0}(p)$, we added q additional ones to each string in $\mathcal{C}_{i_0}$ which increased each Hamming distance by q. Therefore

$$\{H(\mu, \nu_j^{i_0}) : \nu_j^{i_0} \in \mathcal{C}_{i_0}(p)\} = \{(p-7)_{(1)}, (p-6)_{(6)}, (p-5)_{(3)}, (p-4)_{(15)}, \\ (p-3)_{(15)}, (p-2)_{(3)}, (p-1)_{(6)}, p_{(1)}\}.$$

As a result,

$$\mathcal{H}(\mu, \mathcal{C}_{i_0}(p)) = (p-7)!(p-6)!^6(p-5)!^3(p-4)!^{15}(p-3)!^{15}(p-2)!^3(p-1)!^6 p!$$

which is divisible by p. Therefore $\mathcal{H}(\mu) \equiv 0 \mod p$. ∎

Claim 2.22. *For an arbitrary $\mu \in \mathcal{M}(p)$ having Properties 1, 2, and 3, the value*

$$\mathcal{H}(\mu) = (p-6)!^{7k}(p-5)!^{6k}(p-4)!^{12k}(p-3)!^{12k}(p-2)!^{6k+2n}(p-1)!^{7k+2},$$

which is not congruent to 0 modulo p.

Proof. Let $\mu \in \mathcal{M}'_{\Gamma}(p)$. Because it has Property 1,

$$\mathcal{H}(\mu, \mathcal{A}(p)) = (n + (q+4))!^2 = (p-1)!^2.$$

Since μ has Property 2, for any $i \in [n]$,

$$\mathcal{H}(\mu, \mathcal{B}_i(p)) = (n + (q+3))!^2 = (p-2)!^2.$$

Finally, μ satisfies Property 3 which means $\mu \in \mathcal{M}'_{c_i}(p)$ for all clauses c_i in Γ.

Recall that each string $\eta \in \mathcal{C}_i(p)$ is created from a string $\eta' \in \mathcal{C}_i$ by adding q more additional ones. Therefore $H(\mu, \eta) = H(\mu, \eta') + q$. So, the multiset $\mathcal{H}(\mu, \mathcal{C}_i(p))$ can be obtained from $\mathcal{H}(\mu, \mathcal{C}_i)$ found in (2.2) by adding q to each element. As a result,

$$\mathcal{H}(\mu, \mathcal{C}_i(p)) = (p-6)!^7(p-5)!^6(p-4)!^{12}(p-3)!^{12}(p-2)!^6(p-1)!^7.$$

Therefore

$$\mathcal{H}(\mu) = (p-6)!^{7k}(p-5)!^{6k}(p-4)!^{12k}(p-3)!^{12k}(p-2)!^{6k+2n}(p-1)!^{7k+2}. \tag{2.8}$$

Because p is prime, $\mathcal{H}(\mu) \not\equiv 0 \mod p$. ∎

Set

$$T(p) := \sum_{\mu \in \mathcal{M}(p)} \mathcal{H}(\mu).$$

Set $S(p)$ equal to the function of p displayed in (2.8). Thus $S(p)$ is precisely the value of the number of SCJ scenarios admitted by an arbitrary $\mu \in \mathcal{M}'_\Gamma(p)$. If we calculate $T(p) \mod p$, the four claims show that

$$T(p) \equiv \sum_{\mu \in \mathcal{M}'_\Gamma(p)} \mathcal{H}(\mu) \equiv |\mathcal{M}'_\Gamma(p)| \cdot S(p) \mod p. \tag{2.9}$$

If γ is the number of satisfying truth assignments for Γ, then $\gamma = |\mathcal{M}'_\Gamma(p)|$ by Definition 2.3. Therefore

$$\gamma \cdot S(p) \equiv T(p) \mod p.$$

Since p does not divide $S(p)$ (Claim 2.22), there exists an integer $S'(p)$ such that $S(p) \cdot S'(p) \equiv 1 \mod p$. Thus

$$\gamma \equiv S'(p) \cdot T(p) \mod p.$$

While this alone is not sufficient to determine the value of γ, we can repeat this construction for many different prime values to obtain more congruences.

Recall p was fixed to be a prime greater than n' and at most $5n'$. Repeat the above construction for each prime $p_1, p_2, \ldots, p_m$ in this range. The result is a list of congruences:

$$\begin{aligned}
\gamma &\equiv S'(p_1) \cdot T(p_1) \mod p_1, \\
\gamma &\equiv S'(p_2) \cdot T(p_2) \mod p_2, \\
&\vdots \\
\gamma &\equiv S'(p_m) \cdot T(p_m) \mod p_m.
\end{aligned}$$

Because $p_1, p_2, \ldots, p_m$ are all prime, the Chinese Remainder Theorem guarantees a solution for γ which is unique modulo $\prod_{i\in[m]} p_i$. By the Corollary 2.17,

$$\prod_{i\in[m]} p_i = \frac{\prod\limits_{\substack{p\leq 5n' \\ p \text{ prime}}} p}{\prod\limits_{\substack{p\leq n' \\ p \text{ prime}}} p} \geq \frac{e^{5n'/2}}{e^{3n'/2}} = e^{n'} \geq e^n.$$

Since γ is the number of satisfying truth assignments for Γ, and there are only n literals which can realize one of two values, $\gamma \leq 2^n$. Since $\prod_{i\in[m]} p_i \geq e^n > 2^n \geq \gamma$, the Chinese Remainder Theorem gives the exact value of γ.

In summary, for D3CNF Γ with n variables and k clauses, we use the Sieve of Eratosthenes to identify the primes between n' and $5n'$. This runs in $O(n^2)$ time. Then for each prime p in this interval (which is at most $\max\{2n, 600\}$ primes), we create $50k + 2n + 2$ binary strings of length $2n + t(p)$ where $t(p)$ is a polynomial (2.5) in n and p with $p \in O(n)$. Finally, the Chinese Remainder Theorem will solve the system of congruences in $O(\log(p_1 p_2 \ldots p_m))^2$ time (Bach and Shallit 1996). For us, this is $O(n \log n)^2$ because each prime is at most $5n$ and $m \leq 2n$.

Therefore, if we had a polynomial time algorithm to determine the total number of most parsimonious scenarios for a collection of binary strings, then we have created here a polynomial time algorithm to determine the number of satisfying truth assign-

ments for a D3CNF, a problem which is known to be #P-complete. This finishes the proof. ∎

2.3 Torpidly mixing Markov chain

In the previous section, we proved that #StarSPSCJ is a #P-complete problem. The natural next question is whether or not the number of most parsimonious SCJ scenarios can be approximated. More important, can we sample from the most parsimonious SCJ scenarios almost uniformly? In this way, one can test hypotheses on a sample random sample since there are too many most parsimonious SCJ scenarios to test hypotheses on all of them.

Definition 2.23. *A counting problem #A in #P has an FPAUS (fully polynomial almost uniform sampler) if there is a randomized algorithm such that, for any instance of #A and any $\epsilon > 0$, the algorithm outputs an element $x \in X$, the solution space for #A, with probability $p(x)$ where*

$$\frac{1}{2} \sum_{x \in X} |p(x) - U(x)| \leq \epsilon$$

where U is the uniform distribution on X and the algorithm runs in time polynomial in the size of the instance of #A and $-\log \epsilon$.

The technique which was used to prove that #StarSPSCJ is in #P-complete has been used to show that other problems are #P-complete. For example, Brightwell and Winkler (1991) used this technique to prove that counting the number of linear extensions of a partially order set is #P-complete. For this same problem, Karzanov and Khachiyan (1991) found a rapidly mixing Markov chain to sample the linear extensions.This would suggest that #StarSPSCJ also has an FPAUS. However, here we give a straightforward Markov chain to sample the most parsimonious SCJ scenarios that turns out to be torpidly mixing, suggesting that #StarSPSCJ may not have an

FPAUS. With evidence for both the positive answer and the negative answer, the question of whether or not #StarSPSPCJ has an FPAUS remains open.

The Markov chain that we define here is stated for the more general #SPSCJ problem on the star tree T where the multiset of genomes labeling the leaves is not required to have independent adjacencies. Then we prove that this Markov chain is torpidly mixing for a class of instances in which the adjacencies are independent.

2.3.1 Defining the Markov Chain

Fix a multiset $\mathbb{G}$ of genomes to label the leaves of star tree T. Consider the set of median genomes that could label the center of the star in a most parsimonious labeling.

Note that if $\mathbb{G}$ has an odd number of strings, then there is exactly one median. Here we assume that the size of $\mathbb{G}$ is even.

Given the binary string representations $\{\nu_1, \nu_2, \ldots, \nu_m\}$ for the genomes in $\mathbb{G}$, a median μ must minimize $\sum_{i\in[m]} H(\nu_i, \mu)$. Each *majority adjacency*, an adjacency which appears in more than half of the genomes in $\mathbb{G}$, must also appear in every median. Because these appear in more than half of the genomes, for every pair of majority adjacencies, there is a genome in $\mathbb{G}$ that contains both of them. Therefore, containing all majority adjacencies does not conflict with the requirement that a median is a valid genome. If an adjacency appears in fewer than half of the genomes in $\mathbb{G}$, then it must not appear in any median.

Among the *ambiguous adjacencies*, adjacencies that appear in exactly half of the genomes in $\mathbb{G}$, medians may contain any subset of these as long as the result is a valid genome. Each median is characterized by its ambiguous adjacencies. If we draw the adjacency graph A with vertices for the extremities and edges just for the ambiguous adjacencies for $\mathbb{G}$, then the medians are in one-to-one correspondence with the subsets of edges that form a matching (not necessarily maximal and possibly empty).

Here we define a primer Markov chain, P, to transition between the medians. As mentioned, the medians are in one-to-one correspondence with the matchings of A. So it suffices to define our Markov Chain on the state space of matchings in A.

Define the Markov chain P. From any matching $M_0 \subseteq E(A)$ (corresponding to median $\mathcal{M}_0$), we may transition to another matching with the following probabilities:

- With probability 1/2, remain in the current state.
- With probability 1/2, randomly and uniformly choose an edge from A.
  - If the edge is already in matching M_0, then remove it.
    (i.e. If the adjacency is in median $\mathcal{M}_0$, then cut it, replacing the adjacency with two telomeres.)
  - If the edge is not in matching M_0 and adding the edge will extend M_0 to a larger matching in A, then include it.
    (i.e. If the join of two telomeres of $\mathcal{M}_0$ will create the selected adjacency, then join them.)
  - Otherwise, do nothing.

Because we remain at the current state with probability $\frac{1}{2}$, by definition we have created a lazy Markov chain.

Observation 2.24. *The primer Markov chain P is irreducible and aperiodic.*

Proof. Every matching M can be reached from every other matching N by removing all edges from N one at a time and then adding the edges of M one at a time. Each step in this process is completed with positive probability. By definition, the primer Markov chain is irreducible.

Because we have a lazy Markov chain, the probability that we remain at the current state is at least 1/2 and thus the period is 1 for every state. Consequently P is an aperiodic Markov chain. ∎

For two genomes, $\mathcal{G}$ and $\mathcal{M}$, the number of different SCJ scenarios that transform $\mathcal{M}$ into $\mathcal{G}$ is notated by $S(\mathcal{M}, \mathcal{G})$. In particular $S(\mathcal{M}, \mathcal{G})$ is at most $H(M, G)!$ where M and G are the binary string representations of $\mathcal{M}$ and $\mathcal{G}$ respectively. The factorial bound is achieved precisely when $\{\mathcal{M}, \mathcal{G}\}$ has independent adjacencies. For a fixed multiset of genomes $\mathbb{G} = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_m\}$ and median $\mathcal{M}$, the number of scenarios admitted by this median is defined by the function f as

$$f(\mathcal{M}) := \prod_{i=1}^{m} S(\mathcal{M}, \mathcal{G}_i).$$

With this new function and the primer Markov chain P, we employ the Metropolis-Hastings algorithm (Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller 1953) to obtain a secondary Markov chain C with a desired limit distribution. The states remain the same, but the transition probabilities are changed in the following way. From state $\mathcal{M}$, we propose a next state $\mathcal{M}_{new}$ which is reachable from $\mathcal{M}$ in one step as described in the state space for P. If $\mathcal{M}_{new}$ is different from $\mathcal{M}$, accept this transition with probability

$$\min\left\{1, \frac{f(\mathcal{M}_{new})}{f(\mathcal{M})}\right\}.$$

In other words, if $\mathcal{M}_{new}$ was reached from $\mathcal{M}$ with probability $P(\mathcal{M}_{new}|\mathcal{M})$, then in the secondary Markov chain C the transition from $\mathcal{M}$ to $\mathcal{M}_{new}$ will be made with probability

$$C(\mathcal{M}_{new}|\mathcal{M}) = P(\mathcal{M}_{new}|\mathcal{M}) \cdot \min\left\{1, \frac{f(\mathcal{M}_{new})}{f(\mathcal{M})}\right\}.$$

Given the function f on the medians for a given collection of genomes, we obtain a probability distribution θ on these medians by taking $\theta(\mathcal{M})$ to be directly proportional to $f(\mathcal{M})$. In other words,

$$\theta(\mathcal{M}) \propto f(\mathcal{M}) \tag{2.10}$$

or $\theta(\mathcal{M}) = kf(\mathcal{M})$ for some constant k and any median $\mathcal{M}$.

Observation 2.25. *Markov chain C is reversible and converges to the limit distribution θ.*

Proof. A Markov chain is reversible if

$$\theta(\mathcal{M}_i)C(\mathcal{M}_j|\mathcal{M}_i) = \theta(\mathcal{M}_j)C(\mathcal{M}_i|\mathcal{M}_j)$$

where $C(\mathcal{M}_j|\mathcal{M}_i)$ is the probability that state $\mathcal{M}_j$ is reached from state $\mathcal{M}_i$ in one step. This is trivially true if $\mathcal{M}_i$ and $\mathcal{M}_j$ cannot reach one another in one transition step. Otherwise, there is a single edge that can be added or removed from $\mathcal{M}_i$ to obtain $\mathcal{M}_j$. Without loss of generality, we assume $\frac{f(\mathcal{M}_i)}{f(\mathcal{M}_j)} \leq 1 \leq \frac{f(\mathcal{M}_j)}{f(\mathcal{M}_i)}$. Observe the following:

$$\begin{aligned}
\theta(\mathcal{M}_i)C(\mathcal{M}_j|\mathcal{M}_i) &= kf(\mathcal{M}_i)\frac{1}{2}\frac{1}{|E(H)|}\min\left\{1, \frac{f(\mathcal{M}_j)}{f(\mathcal{M}_i)}\right\} \\
&= kf(\mathcal{M}_i)\frac{1}{2|E(H)|} \\
&= k\frac{f(\mathcal{M}_i)}{f(\mathcal{M}_j)}f(\mathcal{M}_j)\frac{1}{2|E(H)|} \\
&= kf(\mathcal{M}_j)\frac{1}{2|E(H)|}\min\left\{1, \frac{f(\mathcal{M}_i)}{f(\mathcal{M}_j)}\right\} \\
&= \theta(\mathcal{M}_j)C(\mathcal{M}_i|\mathcal{M}_j)
\end{aligned}$$

Therefore the Markov chain C is reversible.

As a result of reversibility, we can quickly confirm that $\theta C = \theta$ by the properties of matrix multiplication. Therefore θ is a stationary distribution.

Notice in Markov chain C we can reach any state from any other state. Therefore it is irreducible. In addition, it is finite. Because C is lazy, it is aperiodic. These properties, together with the fact that a stationary distribution exists, imply that C has a limiting distribution which is precisely the stationary distribution. ∎

Therefore, we have a Markov chain on the state space of medians which, in the limit, will sample medians with distribution proportional to the number of scenarios each median admits. Once we have a median, it is easy to uniformly sample from the scenarios that it admits.

Now we will show that the Markov chain C is torpidly mixing (not rapidly mixing). In fact, it is torpidly mixing even when the number of genomes is fixed while the number of genes and adjacencies is allowed to grow. To prove this result, we will use the following:

For any nonempty subset S of the set of medians M, the *capacity of* S is

$$\pi(S) := \sum_{\mu \in S} \pi(\mu)$$

and the *ergodic flow out of* S is

$$F(S) := \sum_{\substack{\mu \in S \\ \nu \in M \setminus S}} \pi(\mu) C(\mu|\nu).$$

The *conductance* is

$$\Phi := \min \left\{ \frac{F(S)}{\pi(S)} : S \subseteq M, 0 < \pi(S) \leq \frac{1}{2} \right\}.$$

Theorem 2.26 (Mélykúti (2006))**.** *A Markov chain is rapidly mixing if and only if* $\Phi \geq \frac{1}{p(n)}$ *for some polynomial* $p(n)$ *which is not identically zero.*

Consider the following example. Select genomes $\mathcal{G}_0$ and $\mathcal{G}_1$ which are represented by matchings G_0 and G_1 in the adjacency graph A. Further, assume that $\{\mathcal{G}_0, \mathcal{G}_1\}$ have independent adjacencies. More specifically, assume $\mathcal{G}_0$ is represented by the empty matching and $\mathcal{G}_1$ is represented by a maximum matching on A. Therefore each median for $\{\mathcal{G}_0, \mathcal{G}_1\}$ will be characterized by a subset of the adjacencies in $\mathcal{G}_1$.

In their binary string representations, it suffices to only consider the coordinates that represent adjacencies in $\mathcal{G}_1$ since these are the only adjacencies of interest. Therefore, we may view $\mathcal{G}_0$ as a binary string with all zeros, $\{0\}^n$, and $\mathcal{G}_1$ as the binary string of all ones, $\{1\}^n$. The set of medians will be $\{0, 1\}^n$.

Consider the star tree with $2t$ leaves, t of which are labeled $\mathcal{G}_0$ and the other t are labeled $\mathcal{G}_1$. First observe that every binary string in $\{0, 1\}^n$ is still a median. Further,

a median μ with exactly k ones admits $(k!(n-k)!)^t$ scenarios. The total number of scenarios, added over all medians, is $T := \sum_{k=0}^{n} \binom{n}{k} (k!(n-k)!)^t$. Therefore

$$\theta(\mu) = \frac{(k!(n-k)!)^t}{T}.$$

Suppose n is odd. Consider the subset S which contains all medians with at most $\left\lfloor \frac{n}{2} \right\rfloor$ ones. For this subset, the capacity is

$$\pi(S) = \frac{1}{2}.$$

For the ergodic flow out of S, we have

$$\begin{aligned}
F(S) &= \sum_{\substack{\mu \in S \\ \nu \in M \setminus S}} \pi(\mu) C(\mu|\nu) \\
&= \sum_{\substack{\mu \in S \\ \nu \in M \setminus S}} \frac{1}{T} \left(\left\lfloor \frac{n}{2} \right\rfloor ! \left\lceil \frac{n}{2} \right\rceil ! \right)^t \frac{1}{2} \frac{1}{n} \cdot 1 \\
&= \frac{1}{2nT} \binom{n}{\lfloor \frac{n}{2} \rfloor} \binom{n}{\lceil \frac{n}{2} \rceil} \left(\left\lfloor \frac{n}{2} \right\rfloor ! \left\lceil \frac{n}{2} \right\rceil ! \right)^t \\
&= \frac{1}{2nT} (n!)^2 \left(\left\lfloor \frac{n}{2} \right\rfloor ! \left\lceil \frac{n}{2} \right\rceil ! \right)^{t-4} \\
&\leq \frac{1}{2n} \frac{1}{(n!)^t} (n!)^2 \left(\left\lfloor \frac{n}{2} \right\rfloor ! \left\lceil \frac{n}{2} \right\rceil ! \right)^{t-4} \\
&= \frac{1}{2n} \frac{1}{(n!)^2} \frac{1}{\left(\binom{n}{\lfloor \frac{n}{2} \rfloor} \right)^{t-4}} \\
&\leq \frac{1}{2n^{t-3}} \frac{1}{(n!)^2}.
\end{aligned}$$

This implies

$$\Phi \leq \frac{F(S)}{\pi(S)} \leq \frac{1}{n^{t-3}(n!)^2}$$

Therefore, if $t \geq 1$, then as n grows, we see that Φ cannot be lower-bounded by a function of the form $\frac{1}{p(n)}$ where p is a polynomial in n. Therefore the Markov chain C is torpidly mixing by Theorem 2.26.

Chapter 3

Generalizations for the Star Tree

In this chapter, we consider a generalization of #StarSPSCJ.

First, fix a continuous function $f : \mathbb{R} \to \mathbb{R}$. Then define the following problem:

Definition 3.1 (#StarSPSCJ(f))**.** *Given an arbitrary $m \in \mathbb{Z}^+$, let $S = \{\nu_i\}_{i=1}^m$ be an arbitrary multiset of binary strings. Determine the value of*

$$\sum_{\mu \in \mathcal{M}(S)} \prod_{i \in [m]} f(H(\nu_i, \mu)).$$

3.1 Calculating #StarSPSCJ(f) exactly

In the previous section, we showed that #StarSPSCJ(f) is #P-complete when $f(x)$ is the function $x!$. Here we work toward the determining the computational complexity of #StarSPSCJ(f) for various functions f. First, we formalize a definition and develop a couple of tools.

Definition 3.2. *A function $g : \mathbb{R} \to \mathbb{R}$ is strictly concave up if for any $x, y, z \in \mathbb{R}$, $x < y < z$,*

$$\frac{g(z) - g(x)}{z - x} > g(y).$$

Equivalently, $g'(x)$ is a strictly increasing function.

Lemma 3.3. *If $\log f(x)$ is a strictly concave up function, then for any $x < y$ and $a > 0$,*

$$\frac{f(x)f(y)}{f(x-a)f(y+a)} < 1.$$

Proof. By the intermediate value theorem, there are real values c, d with $c \in (x-a, x)$ and $d \in (y, y+a)$ such that

$$(\log f)'(c) = \frac{1}{a}(\log f(x) - \log f(x-a)) \text{ and}$$
$$(\log f)'(d) = \frac{1}{a}(\log f(y+a) - \log f(y)).$$

Because $\log f(x)$ is strictly increasing, $g'(c) < g'(d)$. Therefore,

$$\begin{aligned}
\frac{1}{a}(\log f(x) - \log f(x-a)) &< \frac{1}{a}(\log f(y+a) - \log f(y)) \\
\log f(x) - \log f(x-a) &< \log f(y+a) - \log f(y) \\
\log \frac{f(x)}{f(x-a)} &< \log \frac{f(y+a)}{f(y)} \\
\frac{f(x)}{f(x-a)} &< \frac{f(y+a)}{f(y)} \\
\frac{f(x)f(y)}{f(x-a)f(y+a)} &< 1.
\end{aligned}$$

■

Fact 3.4. *Fix $k \in \mathbb{Z}^{\geq 0}$. Let $f(x)$ be a function such that $\log f(x)$ is strictly concave up. Then*

$$\min_{\substack{\alpha,\beta \in \mathbb{Z}^{\geq 0} \\ a+b=k}} f(a)f(b) = f\left(\left\lfloor \frac{k}{2} \right\rfloor\right) f\left(\left\lceil \frac{k}{2} \right\rceil\right).$$

Proof. Let $x = \left\lfloor \frac{k}{2} \right\rfloor$ and $y = \left\lceil \frac{k}{2} \right\rceil$. By Lemma 3.3, $f(x-a)f(y+a) < f(x)f(y)$ which gives the desired result. ■

Theorem 3.5. *Fix a function $f(x) : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ which satisfies the following properties:*

- *$\log f(x)$ is strictly concave up,*
- *the function values of f can be computed in polynomial time, and*

- *for all but finitely many* $n \in \mathbb{Z}$, $n \geq 2$,

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} > 1.$$

For arbitrary $m, s \in \mathbb{Z}^{>0}$ *and* $D \in \mathbb{R}$, *let* $S := \{\nu_1, \nu_2, \ldots, \nu_m\}$ *be a multiset of binary strings, each of length* s. *Then it is #P-complete to determine how many medians* μ *for* S *have*

$$\prod_{i \in [m]} f\left(H(\nu_i, \mu)\right) \leq D. \tag{3.1}$$

Proof. Fix a function $f(x)$ with the properties listed in the theorem.

If is straightforward to see that #StarSPSCJ(f) is in #P. Fix an instance consisting of integer m, s, real number D, and a multiset S of binary strings of length ℓ. Let μ be a binary string of the same length as each ν_i. We can verify that μ is a median in time $O(m\ell)$. Each $H(\nu_i, \mu)$ can be computed in time $O(\ell)$. Because $H(\nu_i, \mu) \leq \ell$, we can compute $f(H(\nu_i, \mu)$ in time polynomial in the size of the input by the conditions on f. Finally, checking if the product is at most D is also a polynomial time calculation. Therefore #StarSPSCJ(f) is in #P.

To prove NP-hardness, we will provide a reduction from #D3SAT. For Γ, a D3CNF with n variables and k clauses, set

$$\begin{aligned}\kappa(n) =& [f(n)]^{9k}[f(n+1)]^{22k+2kn}[f(n+2)]^{48k+12kn} \\ & \cdot [f(n+3)]^{48k+12kn}[f(n+4)]^{22k+2kn}[f(n+5)]^{9k}\end{aligned}$$

The idea is to define a multiset, $\mathcal{D}$, of binary strings with the following properties:

- Each median which corresponds to a satisfying truth assignment for Γ will have

$$\prod_{\eta \in \mathcal{D}} f(H(\eta, \mu)) = \kappa$$

- Each other median will have

$$\prod_{\eta \in \mathcal{D}} f(H(\eta, \mu)) > \kappa.$$

For arbitrary $n, k \in \mathcal{Z}^{>0}$, fix $\Gamma = c_1 \wedge c_2 \wedge \ldots \wedge c_k$ be an arbitrary D3CNF with n variables.

Create a total of $158k + 28kn$ strings of length $2n + 260k + 35kn$ with coordinates

$$(x_1, y_1, x_2, y_2, \ldots, x_n, y_n, e_1, e_2, \ldots, e_t)$$

where $t = 260k + 35kn$.

This multiset of binary strings will be defined as the union of three multisets:

$$\mathcal{D} = \mathcal{A} \uplus \biguplus_{i \in [n]} \mathcal{B}_i \uplus \biguplus_{i \in [k]} \mathcal{C}_i.$$

As in Definition 2.4, we will define each string $\eta \in \mathcal{D}$ by explicitly giving the values of $\eta[x_j]$ and $\eta[y_j]$ for each $i \in [n]$ and telling the number of additional ones in η.

The collection $\mathcal{A}$ contains $108k$ strings. For $a \in [t]$, let $\alpha^{(+a)}$ be the string with $\alpha[x_i] = \alpha[y_i] = 1$ for all $1 \leq i \leq n$ and a additional ones. Define $\overline{\alpha}^{(+a)}$ to be the binary string which is complementary to $\alpha^{(+0)}$ on the first $2n$ coordinates and has a additional ones. The multiset $\mathcal{A}$ will consist of the following strings:

- k copies each of $\alpha^{(+0)}$ and $\overline{\alpha}^{(+0)}$,
- 8k copies each of $\alpha^{(+1)}$ and $\overline{\alpha}^{(+1)}$,
- 18k copies each of $\alpha^{(+2)}$ and $\overline{\alpha}^{(+2)}$,
- 18k copies each of $\alpha^{(+3)}$ and $\overline{\alpha}^{(+3)}$,
- 8k copies each of $\alpha^{(+4)}$ and $\overline{\alpha}^{(+4)}$,
- k copies each of $\alpha^{(+5)}$ and $\overline{\alpha}^{(+5)}$.

The collection $\mathcal{B} = \biguplus_{i\in[n]} \mathcal{B}_i$ contains $28kn$ strings. For each $i \in [n]$, $a \in [t]$, let $\beta_i^{(+a)}$ be the string with $\beta_i[x_i] = \beta_i[y_i] = 1$, $\beta_i[x_j] = \beta_i[y_j] = 0$ for $j \neq i$, and with a additional ones. Define the binary string $\overline{\beta_i}^{(+a)}$ to be complementary to $\beta_i^{(+0)}$ on the first $2n$ coordinates and have a additional ones. The collection $\mathcal{B}_i$ consists of the following $28k$ strings.

- k copies each of $\beta_i^{(+1)}$ and $\overline{\beta_i}^{(+1)}$,
- $6k$ copies each of $\beta_i^{(+2)}$ and $\overline{\beta_i}^{(+2)}$,
- $6k$ copies each of $\beta_i^{(+3)}$ and $\overline{\beta_i}^{(+3)}$,
- k copies each of $\beta_i^{(+4)}$ and $\overline{\beta_i}^{(+4)}$.

The collection $\mathcal{C} = \biguplus_{i\in[k]} \mathcal{C}'_i$ contains $50k$ strings. Each set $\mathcal{C}'_i$, which is associated with clause c_i, consists of 50 strings. In Section 2.1.1, we defined set $\mathcal{C}_i$ through Table 2.1. For each $\nu_j^i \in \mathcal{C}_i$, create $\hat{\nu}_j^i$ by increasing the number of additional ones in ν_j^i by one. Then

$$\mathcal{C}'_i := \{\hat{\nu}_j^i : \nu_j^i \in \mathcal{C}_i\}.$$

Let $\mathcal{M}$ be the set of all medians for $\mathcal{D}$. From Definitions 1.10 and 2.3 and for each clause c_i in Γ, set

$$\mathcal{M} := \mathcal{M}(\mathcal{D}) \qquad \mathcal{M}' := \mathcal{M}'(\mathcal{D})$$
$$\mathcal{M}'_{c_i} := \mathcal{M}'_{c_i}(\mathcal{D}) \qquad \mathcal{M}'_{\Gamma} := \mathcal{M}'_{\Gamma}(\mathcal{D})$$

According to Remark 2.4, all medians μ must have $\mu[e_i] = 0$ for all $i \in [t]$. In $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$, the strings come in pairs where one is complementary to the other on the first $2n$ coordinates. By Fact 2.9, each of the x_i and y_i coordinates are ambiguous. Therefore

$$\mathcal{M} = \{0,1\}^{2n} \times \{0\}^t,$$
$$\mathcal{M}' = \{01,10\}^n \times \{0\}^t.$$

Define

$$\mathcal{H}(\mu, \mathcal{A}) := \prod_{a \in \mathcal{A}} f(H(\mu, a)).$$

Similarly define $\mathcal{H}(\mu, \mathcal{B}_i)$ and $\mathcal{H}(\mu, \mathcal{C}'_j)$. Set

$$\mathcal{H}(\mu) := \mathcal{H}(\mu, \mathcal{A}) \cdot \prod_{i \in [n]} \mathcal{H}(\mu, \mathcal{B}_i) \cdot \prod_{j \in [k]} \mathcal{H}(\mu, \mathcal{C}'_j).$$

For each $\mu \in \mathcal{M}$, we obtain a lower bound for $\mathcal{H}(\mu)$ and for each $\mu \in \mathcal{M}'_\Gamma$ we describe an exact value for $\mathcal{H}(\mu)$. Divide $\mathcal{M}$ into 4 classes using the following three properties which a median $\mu \in \mathcal{M}$ may have.

Property 1. $\sum_{i \in [n]} (\mu[x_i] + \mu[y_i]) = n$.

Property 2. $\mu \in \mathcal{M}'$.

Property 3. $\mu \in \mathcal{M}'_\Gamma$.

Notice that these properties are nested. Any median $\mu \in \mathcal{M}$ with Property 2, must also have Property 1. Further, any $\mu \in \mathcal{M}$ with Property 3 must also have Property 2. The following claims provide lower bounds for medians according to their properties.

Claim 3.6. *If $\mu \in \mathcal{M}$ has Property 1, then*

$$\begin{aligned} \mathcal{H}(\mu, \mathcal{A}) =& [f(n)]^{2k} [f(n+1)]^{16k} [f(n+2)]^{36k} \\ & \cdot [f(n+3)]^{36k} [f(n+4)]^{16k} [f(n+5)]^{2k} \qquad (3.2) \\ =:& \alpha_{good}. \end{aligned}$$

Otherwise,

$$\begin{aligned} \mathcal{H}(\mu, \mathcal{A}) \geq& [f(n-1)f(n+1)]^k [f(n)f(n+2)]^{8k} [f(n+1)f(n+3)]^{18k} \\ & \cdot [f(n+2)f(n+4)]^{18k} [f(n+3)f(n+5)]^{8k} [f(n+4)f(n+6)]^k \qquad (3.3) \\ =:& \alpha_{bad}. \end{aligned}$$

Proof. If $\mu \in \mathcal{M}$ has Property 1, then $H(\mu, \alpha^{(+0)}) = H(\mu, \overline{\alpha}^{(+0)}) = n$ because $\alpha^{(+0)}[x_i] = \alpha^{(+0)}[y_i] = 1$ for all $i \in [n]$ while μ only has n ones in the first 2n entries. Because $\mu[e_i] = 0$ for all $i \in [t]$, by Definition 2.4,

$$H(\mu, \alpha^{(+a)}) = H(\mu, \overline{\alpha}^{(+a)}) = n + a.$$

Recalling the exact strings that appear in $\mathcal{A}$, we quickly obtain (3.2).

If μ does not have Property 1, then either μ has more than n ones in the first $2n$ entries, implying $H(\mu, \alpha^{(+0)}) > n$, or μ has less than n ones in the first $2n$ entries, implying $H(\mu, \overline{\alpha}^{(+0)}) > n$. By Fact 2.7, $H(\mu, \alpha^{(+0)}) + H(\mu, \overline{\alpha}^{(+0)}) = 2n$. By Fact 3.4 and the above observations,

$$\begin{aligned} f\left(H\left(\mu, \alpha^{(+0)}\right)\right) \cdot f\left(H(\mu, \overline{\alpha}^{(+0)})\right) &\geq f(n-1)f(n+1), \\ f\left(H\left(\mu, \alpha^{(+a)}\right)\right) \cdot f\left(H\left(\mu, \overline{\alpha}^{(+a)}\right)\right) &\geq f(n-1+a)f(n+1+a). \end{aligned}$$

Recalling the exact strings in $\mathcal{B}$, we obtain the lower bound in (3.3). ∎

Claim 3.7. *For each $\mu \in \mathcal{M}$ and each $i \in [n]$,*

$$\mathcal{H}(\mu, \mathcal{B}_i) \geq [f(n+1)]^{2k}[f(n+2)]^{12k}[f(n+3)]^{12k}[f(n+4)]^{2k} =: \beta_{good}. \tag{3.4}$$

If μ has Property 2, then for every $i \in [n]$,

$$\mathcal{H}(\mu, \mathcal{B}_i) = \beta_{good}.$$

If μ satisfies Property 1, but not Property 2, then there exists $i_0 \in [n]$ such that

$$\begin{aligned} \mathcal{H}(\mu, \mathcal{B}_{i_0}) =& [f(n-1)f(n+3)]^k[f(n)f(n+4)]^{6k} \\ & \cdot [f(n+1)f(n+5)]^{6k}[f(n+2)f(n+6)]^k \\ =:& \beta_{bad}. \end{aligned} \tag{3.5}$$

Proof. For any $\mu \in \mathcal{M}$, by Fact 2.7,

$$H(\mu, \beta^{(+0)}) + H(\mu, \overline{\beta}^{(+0)}) = 2n.$$

By Fact 3.4, for each $a \in \mathbb{Z}^{\geq 0}$,

$$f\left(H\left(\mu, \beta^{(+0)}\right)\right) \cdot f\left(H\left(\mu, \overline{\beta}^{(+0)}\right)\right) \geq [f(n)]^2, \text{ and}$$
$$f\left(H\left(\mu, \beta^{(+a)}\right)\right) \cdot f\left(H\left(\mu, \overline{\beta}^{(+a)}\right)\right) \geq [f(n+a)]^2.$$

Therefore, for any $\mu \in \mathcal{M}$,

$$\mathcal{H}(\mu, \mathcal{B}_i) \geq \beta_{good}.$$

If $\mu \in \mathcal{M}'$, then for each $i \in [n]$, $\mu[x_i] \neq \mu[y_i]$. On the other hand, for each $i \in [n]$, $j \in [n]$, $\beta_i^{(+a)}[x_j] = \beta_i^{(+a)}[y_j]$. Therefore for any $i, j \in [n]$,

$$H((\mu[x_j], \mu[y_j]), (\beta_i^{(+a)}[x_j], \beta_i^{(+a)}[y_j])) = 1.$$

The same holds if $\beta_i^{(+a)}$ is replaced with $\overline{\beta}_i^{(+a)}$. Therefore,

$$H\left(\mu, \beta_i^{(+0)}\right) = H\left(\mu, \overline{\beta}_i^{(+0)}\right) = n,$$
$$H\left(\mu, \beta_i^{(+a)}\right) = H\left(\mu, \overline{\beta}_i^{(+a)}\right) = n + a.$$

As a result $\mathcal{H}(\mu) = \beta_{good}$.

If μ satisfies Property 1 but not Property 2, then we can define a tighter lower bound on $\mathcal{H}(\mu, \mathcal{B}_i)$. In particular, because $\mu \notin \mathcal{M}'$, there exists $i_0 \in [n]$ such that $\mu[x_{i_0}] = \mu[y_{i_0}]$. Recall $\beta_i^{(+a)}[x_{i_0}] = \beta_{i_0}^{(+a)}[y_{i_0}] = 1$ and $\overline{\beta}_{i_0}^{(+a)}[x_{i_0}] = \overline{\beta}_{i_0}^{(+a)}[y_{i_0}] = 0$. Therefore,

$$\begin{aligned}
\mu[x_{i_0}] = 1 \Rightarrow & H((\mu[x_{i_0}], \mu[y_{i_0}]), (\beta_{i_0}^{(+a)}[x_{i_0}], \beta_{i_0}^{(+a)}[y_{i_0}])) = 0, \\
& H((\mu[x_{i_0}], \mu[y_{i_0}]), (\overline{\beta}_{i_0}^{(+a)}[x_{i_0}], \overline{\beta}_{i_0}^{(+a)}[y_{i_0}])) = 2, \text{ and} \\
\mu[x_{i_0}] = 0 \Rightarrow & H((\mu[x_{i_0}], \mu[y_{i_0}]), (\overline{\beta}_{i_0}^{(+a)}[x_{i_0}], \overline{\beta}_{i_0}^{(+a)}[y_{i_0}])) = 0, \\
& H((\mu[x_{i_0}], \mu[y_{i_0}]), (\beta_{i_0}^{(+a)}[x_{i_0}], \beta_{i_0}^{(+a)}[y_{i_0}])) = 2.
\end{aligned}$$

Because μ satisfies Property 1, there are exactly n ones among the first $2n$ coordinates. Without loss of generality, $\mu[x_{i_0}] = \mu[y_{i_0}] = 1$. Set

$$S := \{x_j, y_j : j \in [n], j \neq i_0\}.$$

Then μ has $n-2$ ones and n zeros among the coordinates in S. However, $\beta_{i_0}^{(+a)}$ takes the value 0 on each of the coordinates of S and $\overline{\beta}_{i_0}^{(+a)}$ takes the value 1 on the coordinates of S. Therefore,

$$\begin{aligned}
\mu[x_{i_0}] = 1 \Rightarrow & H(\mu, \beta_{i_0}^{(+0)}) = 0 + (n-2), \\
& H(\mu, \overline{\beta}_{i_0}^{(+0)}) = 2 + n, \text{ and} \\
\mu[x_{i_0}] = 0 \Rightarrow & H(\mu, \overline{\beta}_{i_0}^{(+0)}) = 0 + (n-2), \\
& H(\mu, \beta_{i_0}^{(+0)}) = 2 + n.
\end{aligned}$$

As a result,

$$\begin{aligned}
H(\mu, \beta_{i_0}^{(+0)}) H(\mu, \overline{\beta}_{i_0}^{(+0)}) &= (n-2)(n+2), \\
H(\mu, \beta_{i_0}^{(+a)}) H(\mu, \overline{\beta}_{i_0}^{(+a)}) &= (n-2+a)(n+2+a).
\end{aligned}$$

Taking into account all binary strings in $\mathcal{B}_{i_0}$, we conclude $\mathcal{H}(\mu, \mathcal{B}_{i_0}) = \beta_{bad}$ in (3.5). ∎

Fact 3.8. *For the quantities defined in Claim 3.7, $\beta_{good} < \beta_{bad}$. Consequently, if $\mu \in \mathcal{M} \setminus \mathcal{M}'$ and satisfies Property 1, then $\prod_{i\in[n]} \mathcal{H}(\mu, \mathcal{B}_i) \geq \beta_{bad}\beta_{good}^{k-1}$. If $\mu \in \mathcal{M} \setminus \mathcal{M}'$ and does not satisfy Property 1, then $\prod_{i\in[n]} \mathcal{H}(\mu, \mathcal{B}_i) \geq \beta_{good}^{k}$.*

Proof. Observe

$$\begin{aligned}
\frac{\beta_{bad}}{\beta_{good}} &= \left[\frac{f(n-1)f(n)^6 f(n+1)^4 f(n+4)^4 f(n+5)^6 f(n+6)}{f(n+2)^{11} f(n+3)^{11}}\right]^k \\
&= \left[\frac{f(n-1)f(n+6)}{f(n+2)f(n+3)}\right]^k \cdot \left[\frac{f(n)f(n+5)}{f(n+2)f(n+3)}\right]^{6k} \left[\frac{f(n+1)f(n+4)}{f(n+2)f(n+3)}\right]^{4k} \\
&> 1
\end{aligned}$$

where the last inequality follows from Lemma 3.3. ∎

Claim 3.9. *For any $\mu \in \mathcal{M}$ and for each $j \in [k]$,*

$$\mathcal{H}(\mu, \mathcal{C}_i') \geq [f(n+2)]^{25}[f(n+3)]^{25} =: \gamma_{min}.$$

If $\mu \in \mathcal{M}_\Gamma'$, then for each $j \in [k]$,

$$\mathcal{H}(\mu, \mathcal{C}_i') = [f(n)]^7[f(n+1)]^6[f(n+2)]^{12}[f(n+3)]^{12}[f(n+4)]^6[f(n+5)]^7 =: \gamma_{good}. \tag{3.6}$$

If $\mu \in \mathcal{M}' \setminus \mathcal{M}_\Gamma'$, then there exists $i_0 \in [k]$ such that

$$\begin{aligned} \mathcal{H}(\mu, \mathcal{C}_{i_0}') =& f(n-1)[f(n)]^6[f(n+1)]^3[f(n+2)]^{15} \\ & \cdot [f(n+3)]^{15}[f(n+4)]^3[f(n+5)]^6 f(n+6) \\ =& : \gamma_{bad}. \end{aligned} \tag{3.7}$$

Proof. Let μ be an arbitrary median in $\mathcal{M}$. By Remark 2.12, the binary strings in $\mathcal{C}_i$ come in pairs that are complementary on the first $2n$ entries. With a careful examination of Table 2.1, if $\eta, \eta' \in \mathcal{C}_i$ are complementary on the first $2n$ coordinates, then $e(\eta)+e(\eta') = 3$ where e is the function specifying the number of additional ones. By the definition of $\mathcal{C}_i'$, the strings still come in complementary pairs, $(\hat{\eta}, \hat{\eta}')$, but here $e(\hat{\eta}) + e(\hat{\eta}') = 5$ because the number of additional ones in $\hat{\eta}$ and $\hat{\eta}'$ is precisely one more than the number in η and η'. By Fact 2.7, for each of the 25 pairs in $\mathcal{C}_i'$,

$$H(\mu, \hat{\eta}) + H(\mu, \hat{\eta}') = 2n + 5.$$

Then by Fact 3.4,

$$f(H(\mu, \hat{\eta}))f(H(\mu, \hat{\eta}')) \geq f(n+2)f(n+3)$$

which gives the general bound γ_{min}.

Now suppose $\mu \in \mathcal{M}'_\Gamma$. This implies $\mu \in \mathcal{M}'_{c_i}$ for all clauses c_i in Γ. By the definition of $\mathcal{C}'_i$, for each $\hat{\nu}^i_j \in \mathcal{C}'_i$, $H(\mu, \hat{\nu}^i_j) = H(\mu, \nu^i_j) + 1$ where $\nu^i_j \in \mathcal{C}_i$. From (2.2), we see

$$\{H(\mu, \hat{\nu}^i_j) : j \in [50]\} = \{n_{(7)}, (n+1)_{(6)}, (n+2)_{(12)}, (n+3)_{(12)}, (n+4)_{(6)}, (n+5)_{(7)}\}.$$

This immediately implies $\mathcal{H}(\mu, \mathcal{C}'_i) = \gamma_{good}$ in (3.6).

Finally, suppose $\mu \in \mathcal{M}' \setminus \mathcal{M}'_\Gamma$. Using the bijection in Definition 2.1, μ must correspond to a truth assignment which does not satisfy Γ. So there is a clause c_{i_0} in Γ which is not satisfied. Therefore $\mu \in \mathcal{M}' \setminus \mathcal{M}'_{c_{i_0}}$. From (2.1), adding 1 to each $H(\mu, \nu^{i_0}_j)$ to obtain $H(\mu, \hat{\nu}^{i_0}_j)$, we obtain

$$\{H(\mu, \nu^{i_0}_j) : j \in [50]\} = \{(n-1)_{(1)}, n_{(6)}, (n+1)_{(3)}, (n+2)_{(15)}, (n+3)_{(15)}, (n+4)_{(3)}, (n+5)_{(6)}, (n+6)_{(1)}\}. \tag{3.8}$$

This directly implies $\mathcal{H}(\mu, \mathcal{C}_{i_0}) = \gamma_{bad}$ in (3.7). ∎

Fact 3.10. *For the quantities defined in Claim 3.9, $\gamma_{good} < \gamma_{bad}$. As a result, when $\mu \in \mathcal{M}' \setminus \mathcal{M}'_\Gamma$,*

$$\mathcal{H}(\mu, \mathcal{C}) \geq \gamma_{bad} \gamma^{k-1}_{good}.$$

Proof. Indeed, this was our initial assumption:

$$\frac{\gamma_{bad}}{\gamma_{good}} = \frac{f(n-1)[f(n+2)]^3[f(n+3)]^3 f(n+6)}{f(n)[f(n+1)]^3[f(n+4)]^3[f(n+5)]} > 1.$$

The bound for $\mathcal{H}(\mu, \mathcal{C})$ results from the fact that $\mu \in \mathcal{M}'$ either corresponds to a satisfying truth assignment for c_i or a non-satisfying truth assignment for each clause c_i. ∎

In summary, Claims 3.6, 3.7, and 3.9 along with Facts 3.8 and 3.10, we give the following bounds. If $\mu \in \mathcal{M}'_\Gamma$,

$$\mathcal{H}(\mu) = \alpha_{good} \beta^n_{good} \gamma^k_{good} =: h_3.$$

If $\mu \in \mathcal{M}' \setminus \mathcal{M}'_\Gamma$,

$$\mathcal{H}(\mu) \geq \alpha_{good}\beta_{good}^n \gamma_{bad}\gamma_{good}^{k-1} =: h_2.$$

If $\mu \in \mathcal{M} \setminus \mathcal{M}'$ and has Property 1,

$$\mathcal{H}(\mu) \geq \alpha_{good}\beta_{bad}\beta_{good}^{n-1}\gamma_{min}^k =: h_1.$$

If $\mu \in \mathcal{M}$ but does not have Property 1,

$$\mathcal{H}(\mu) \geq \alpha_{bad}\beta_{good}^n \gamma_{min}^k =: h_0.$$

In order to complete, the proof, we only need to show $h_3 < h_i$ for $i \in \{0, 1, 2\}$. By one of our assumptions about $f(x)$, we have already verified in Fact 3.10 that

$$\frac{h_2}{h_3} = \frac{\gamma_{bad}}{\gamma_{good}} > 1.$$

Next observe

$$\begin{aligned}
\frac{h_1}{h_2} =& \frac{\beta_{bad}}{\beta_{good}} \cdot \frac{\gamma_{min}^k}{\gamma_{bad}\gamma_{good}^{k-1}} \\
>& \frac{\beta_{bad}}{\beta_{good}} \cdot \frac{\gamma_{min}^k}{\gamma_{bad}^k} \\
=& \left[\frac{f(n-1)f(n+3)}{[f(n+1)]^2}\left[\frac{f(n)f(n+4)}{[f(n+2)]^2}\right]^6\left[\frac{f(n+1)f(n+5)}{[f(n+3)]^2}\right]^6\frac{f(n+2)f(n+6)}{[f(n+4)]^2}\right]^k \\
& \cdot \left[\frac{[f(n+2)]^{10}[f(n+3)]^{10}}{f(n-1)[f(n)]^6[f(n+1)]^3[f(n+4)]^3[f(n+5)]^6 f(n+6)}\right]^k \\
=& \left[\frac{f(n+1)f(n+4)}{f(n+2)f(n+3)}\right]^k \\
>& 1
\end{aligned}$$

where the last inequality follows from Lemma 3.3.

Finally we prove that $h_0 > h_2$.

$$\begin{aligned}
\frac{h_0}{h_2} =& \frac{\alpha_{bad}}{\alpha_{good}} \frac{\gamma_{min}^k}{\gamma_{bad}\gamma_{good}^{k-1}} \\
>& \frac{\alpha_{bad}}{\alpha_{good}} \cdot \frac{\gamma_{min}^k}{\gamma_{bad}^k} \\
=& \left[\frac{f(n-1)f(n+1)}{[f(n)]^2}\left[\frac{f(n)f(n+2)}{[f(n+1)]^2}\right]^8\left[\frac{f(n+1)f(n+3)}{[f(n+2)]^2}\right]^{18}\right. \\
& \cdot \left[\frac{f(n+2)f(n+4)}{[f(n+3)]^2}\right]^{18}\left[\frac{f(n+3)f(n+5)}{[f(n+4)]^2}\right]^8\left.\frac{f(n+4)f(n+6)}{[f(n+5)]^2}\right]^k \\
& \cdot \left[\frac{[f(n+2)]^{10}[f(n+3)]^{10}}{f(n-1)[f(n)]^6[f(n+1)]^3[f(n+4)]^3[f(n+5)]^6 f(n+6)}\right]^k \\
=& \frac{[f(n-1)]^k[f(n)]^{8k}[f(n+1)]^{19k}[f(n+2)]^{26k}}{[f(n)]^{2k}[f(n+1)]^{16k}[f(n+2)]^{36k}} \\
& \cdot \frac{[f(n+3)]^{26k}[f(n+4)]^{19k}[f(n+5)]^{8k}[f(n+6)]^k}{[f(n+3)]^{36k}[f(n+4)]^{16k}[f(n+5)]^{2k}} \\
& \cdot \frac{[f(n+2)]^{10k}[f(n+3)]^{10k}}{[f(n-1)]^k[f(n)]^{6k}[f(n+1)]^{3k}[f(n+4)]^{3k}[f(n+5)]^{6k}f(n+6)^k} \\
=& 1.
\end{aligned}$$

Therefore for any $\hat{\mu} \in \mathcal{M}'_\Gamma$ and $\mu \in \mathcal{M} \setminus \mathcal{M}'_\Gamma$, then $\mathcal{H}(\hat{\mu}) < \mathcal{H}(\mu)$. Thus, if we could determine, in polynomial time, how many medians $\mu \in \mathcal{M}$ have $h(\mu) \leq h_3$, then we could determine how many satisfying truth assignments exist for Γ in polynomial time. ∎

Corollary 3.11. *Fix a function $f(x) : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ which satisfies the following properties:*

- $\log f(x)$ *is strictly concave up,*
- *the function values of f can be computed in polynomial time, and*
- *for all but finitely many $n \in \mathbb{Z}$, $n \geq 2$,*

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} > 1.$$

For arbitrary $m, s \in \mathbb{Z}^{>0}$ and $D \in \mathbb{R}$, let $S := \{\nu_1, \nu_2, \ldots, \nu_m\}$ be a multiset of binary strings, each of length s and let $\mathcal{M}$ be the set of medians for S. Then it is NP-complete to determine if

$$\min_{\mu \in \Gamma} \prod_{i \in [m]} f\left(H(\nu_i, \mu)\right) \leq D. \tag{3.9}$$

This next theorem gives the same result as Theorem 3.5 with one change in the conditions on f. While Theorem 3.5 required that

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} > 1,$$

Theorem 3.12 switches the inequality to consider functions in which the ratio is less than 1.

Theorem 3.12. *Fix a function $f(x) : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ which satisfies the following properties:*

- $\log f(x)$ *is strictly concave up,*
- *the function values of f can be computed in polynomial time, and*
- *for all but finitely many $n \in \mathbb{Z}$, $n \geq 2$,*

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} < 1.$$

For arbitrary $m, s \in \mathbb{N}$ and $D \in \mathbb{R}$, let $S := \{\nu_1, \nu_2, \ldots, \nu_m\}$ be a multiset of binary strings, each of length s. Then it is #P-complete to determine how many medians μ for S have

$$\prod_{i \in [m]} f\left(H(\nu_i, \mu)\right) \leq D.$$

Proof. This proof closely mirrors the proof of Theorem 3.5. Here we will note the changes that need to be made.

This time, we define $98k + 24kn$ binary strings, each of length $2n + 245k + 60kn$ with coordinates

$$(x_1, y_1, \ldots, x_n, y_n, e_1, \ldots, e_t).$$

Let $\alpha^{(+a)}$ and $\overline{\alpha}^{(+a)}$ be defined as before. The collection $\mathcal{A}$ will now consist of the following $72k$ strings:

- $4k$ copies each of $\alpha^{(+1)}$ and $\overline{\alpha}^{(+1)}$,
- $14k$ copies each of $\alpha^{(+2)}$ and $\overline{\alpha}^{(+2)}$,
- $14k$ copies each of $\alpha^{(+3)}$ and $\overline{\alpha}^{(+3)}$,
- $4k$ copies each of $\alpha^{(+4)}$ and $\overline{\alpha}^{(+4)}$.

Define $\beta_i^{(+a)}$ and $\overline{\beta}_i^{(+a)}$ as before. The collection $\mathcal{B}_i$ now consists of the following $24k$ strings:

- $6k$ copies each of $\beta_i^{(+2)}$ and $\overline{\beta}_i^{(+2)}$,
- $6k$ copies each of $\beta_i^{(+3)}$ and $\overline{\beta}_i^{(+3)}$.

Following the explanation found in Section 2.1.1, Table 3.1 defines 26 binary strings $\overline{\mathcal{C}_i}$ for a clause. As in the proof of Theorem 3.5, we will add 1 additional one to each of the 26 strings in $\overline{\mathcal{C}_i}$ to create $\mathcal{C}_i'$.

Using the same Properties 1, 2, and 3 as before, we obtain the following values which are analogous to the bounds in Claims 3.6, 3.7, and 3.9:

$$\begin{aligned}
\alpha_{good} :=& [f(n+1)]^{8k}[f(n+2)]^{28k}[f(n+3)]^{28k}[f(n+4)]^{8k}, \\
\alpha_{bad} :=& [f(n)f(n+2)]^{4k}[f(n+1)f(n+3)]^{14k} \\
& \cdot [f(n+2)f(n+4)]^{14k}[f(n+3)f(n+5)]^{4k},
\end{aligned}$$

$$\begin{aligned}
\beta_{good} :=& [f(n+2)]^{12k}[f(n+3)]^{12k},\\
\beta_{min} :=& [f(n+2)]^{12k}[f(n+3)]^{12k},\\
\beta_{bad} :=& [f(n)f(n+4)]^{6k}[f(n+1)f(n+5)]^{6k},\\
\gamma_{good} :=& f(n-1)[f(n)]^3[f(n+1)]^3[f(n+2)]^6\\
&\cdot [f(n+3)]^6[f(n+4)]^3[f(n+5)]^3 f(n+6),\\
\gamma_{bad} :=& [f(n)]^4[f(n+1)]^6[f(n+2)]^3[f(n+3)]^3[f(n+4)]^6[f(n+5)]^4,\\
\gamma_{min} :=& [f(n+2)]^{13}[f(n+3)]^{13}.
\end{aligned}$$

By our assumption about $f(x)$,

$$\frac{\gamma_{bad}}{\gamma_{good}} = \frac{f(n)[f(n+1)]^3[f(n+4)]^3 f(n+5)}{f(n-1)[f(n+2)]^3[f(n+3)]^3 f(n+6)} > 1$$

which implies $\gamma_{bad} > \gamma_{good}$.

Next we determine the values of h_0, h_1, h_2, h_3 in this setting. As before, if μ has Property i, but not property $i+1$, then $\mathcal{H}(\mu) \geq h_i$. Further, if μ has Property 3, $\mathcal{H}(\mu) = h_3$. If μ does not have Property 1, then $\mathcal{H}(\mu) \geq h_0$.

$$\begin{aligned}
h_3 &:= \alpha_{good}\beta_{good}^n\gamma_{good}^k.\\
h_2 &:= \alpha_{good}\beta_{good}^n\gamma_{bad}\gamma_{good}^{k-1}.\\
h_1 &:= \alpha_{good}\beta_{bad}\beta_{min}^{n-1}\gamma_{min}^k.\\
h_0 &:= \alpha_{bad}\beta_{min}^n\gamma_{min}^k.
\end{aligned}$$

As in the proof of Theorem 3.5, we will show $h_3 < h_0, h_1, h_2$.

By our assumption about $f(x)$,

$$\frac{h_2}{h_3} = \frac{\gamma_{bad}}{\gamma_{good}} > 1.$$

Also

$$\begin{aligned}
\frac{h_1}{h_2} =& \frac{\beta_{bad}}{\beta_{good}} \cdot \frac{\gamma_{min}^k}{\gamma_{bad}\gamma_{good}^{k-1}} \\
>& \frac{\beta_{bad}}{\beta_{good}} \cdot \frac{\gamma_{min}^k}{\gamma_{bad}^k} \\
=& \left[\left[\frac{f(n)f(n+4)}{[f(n+2)]^2}\right]^6 \left[\frac{f(n+1)f(n+5)}{[f(n+3)]^2}\right]^6\right]^k \\
& \cdot \left[\frac{[f(n+2)]^{10}[f(n+3)]^{10}}{[f(n)]^4[f(n+1)]^6[f(n+4)]^6[f(n+5)]^4}\right]^k \\
=& \left[\frac{f(n)f(n+5)}{f(n+2)f(n+3)}\right]^{2k} \\
>& 1 \qquad \text{by Lemma 3.3.}
\end{aligned}$$

Finally we show $h_0 > h_2$.

$$\begin{aligned}
\frac{h_0}{h_2} =& \frac{\alpha_{bad}}{\alpha_{good}} \frac{\gamma_{min}^k}{\gamma_{bad}\gamma_{good}^{k-1}} \\
>& \frac{\alpha_{bad}}{\alpha_{good}} \cdot \frac{\gamma_{min}^k}{\gamma_{bad}^k} \\
=& \left[\left[\frac{f(n)f(n+2)}{[f(n+1)]^2}\right]^4 \left[\frac{f(n+1)f(n+3)}{[f(n+2)]^2}\right]^{14}\right. \\
& \left. \cdot \left[\frac{f(n+2)f(n+4)}{[f(n+3)]^2}\right]^{14} \left[\frac{f(n+3)f(n+5)}{[f(n+4)]^2}\right]^4\right]^k \\
& \cdot \left[\frac{[f(n+2)]^{10}[f(n+3)]^{10}}{[f(n)]^4[f(n+1)]^6[f(n+4)]^6[f(n+5)]^4}\right]^k \\
=& \frac{[f(n)]^{4k}[f(n+1)]^{14k}[f(n+2)]^{18k}[f(n+3)]^{18k}[f(n+4)]^{14k}[f(n+5)]^{4k}}{[f(n+1)]^{8k}[f(n+2)]^{28k}[f(n+3)]^{28k}[f(n+4)]^{8k}} \\
& \cdot \frac{[f(n+2)]^{10k}[f(n+3)]^{10k}}{[f(n)]^{4k}[f(n+1)]^{6k}[f(n+4)]^{6k}[f(n+5)]^{4k}} \\
=& 1.
\end{aligned}$$

Making each of these changes in the proof of Theorem 3.5, we complete the proof of Theorem 3.12. ∎

Table 3.1 The 26 strings to complement the collection in Table 2.1 along with their Hamming distance from medians in $\mathcal{M}'$.

	A	**B**	**C**	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**	**M8**
		$\nu_j^i[x_\ell]$,		10 10 10	10 10 01	10 01 10	01 10 10	10 01 01	01 10 01	01 01 10	01 01 01
Row #	Values of ν_j^i on its support set	$\nu_j^i[y_\ell]$ ($v_\ell \notin c_i$)	Add'l Ones								
1	01 00 00	0	+0	$n+1$	$n+1$	$n+1$	$n-1$	$n+1$	$n-1$	$n-1$	$n-1$
2	00 01 00	0	+0	$n+1$	$n+1$	$n-1$	$n+1$	$n-1$	$n+1$	$n-1$	$n-1$
3	00 00 01	0	+0	$n+1$	$n-1$	$n+1$	$n+1$	$n-1$	$n-1$	$n+1$	$n-1$
4	10 11 11	1	+3	$n+2$	$n+2$	$n+2$	$n+4$	$n+2$	$n+4$	$n+4$	$n+4$
5	11 10 11	1	+3	$n+2$	$n+2$	$n+4$	$n+2$	$n+4$	$n+2$	$n+4$	$n+4$
6	11 11 10	1	+3	$n+2$	$n+4$	$n+2$	$n+2$	$n+4$	$n+4$	$n+2$	$n+4$
7	01 01 00	0	+2	$n+4$	$n+4$	$n+2$	$n+2$	$n+2$	$n+2$	n	n
8	01 00 01	0	+2	$n+4$	$n+2$	$n+4$	$n+2$	$n+2$	n	$n+2$	n
9	00 01 01	0	+2	$n+4$	$n+2$	$n+2$	$n+4$	n	$n+2$	$n+2$	n
10	10 10 11	1	+1	$n-1$	$n-1$	$n+1$	$n+1$	$n+1$	$n+1$	$n+3$	$n+3$
11	10 11 10	1	+1	$n-1$	$n+1$	$n-1$	$n+1$	$n+1$	$n+3$	$n+1$	$n+3$
12	11 10 10	1	+1	$n-1$	$n+1$	$n+1$	$n-1$	$n+3$	$n+1$	$n+1$	$n+3$
13	10 01 01	0	+1	$n+2$	n	n	$n+4$	$n-2$	$n+2$	$n+2$	n
14	01 10 01	0	+1	$n+2$	n	$n+4$	n	$n+2$	$n-2$	$n+2$	n
15	01 01 10	0	+1	$n+2$	$n+4$	n	n	$n+2$	$n+2$	$n-2$	n
16	10 10 01	0	+1	n	$n-2$	$n+2$	$n+2$	n	n	$n+4$	$n+2$
17	10 01 10	0	+1	n	$n+2$	$n-2$	$n+2$	n	$n+4$	n	$n+2$
18	01 10 10	0	+1	n	$n+2$	$n+2$	$n-2$	$n+4$	n	n	$n+2$
19	10 10 10	0	+1	$n-2$	n	n	n	$n+2$	$n+2$	$n+2$	$n+4$
20	01 01 01	1	+2	$n+5$	$n+3$	$n+3$	$n+3$	$n+1$	$n+1$	$n+1$	$n-1$
21	10 01 01	1	+2	$n+3$	$n+1$	$n+1$	$n+5$	$n-1$	$n+3$	$n+3$	$n+1$
22	01 10 01	1	+2	$n+3$	$n+1$	$n+5$	$n+1$	$n+3$	$n-1$	$n+3$	$n+1$
23	01 01 10	1	+2	$n+3$	$n+5$	$n+1$	$n+1$	$n+3$	$n+3$	$n-1$	$n+1$
24	10 10 01	1	+2	$n+1$	$n-1$	$n+3$	$n+3$	$n+1$	$n+1$	$n+5$	$n+3$
25	10 01 10	1	+2	$n+1$	$n+3$	$n-1$	$n+3$	$n+1$	$n+5$	$n+1$	$n+3$
26	01 10 10	1	+2	$n+1$	$n+3$	$n+3$	$n-1$	$n+5$	$n+1$	$n+1$	$n+3$

The information in this table is to be read in the same way as the information in Table 2.1. This is detailed in Section 2.1.1.

Corollary 3.13. *Fix a function $f(x) : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ which satisfies the following properties:*

- $\log f(x)$ *is strictly concave up,*
- *the function values of f can be computed in polynomial time, and*
- *for all but finitely many $n \in \mathbb{Z}$, $n \geq 2$,*

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} < 1.$$

For arbitrary $m, s \in \mathbb{Z}^{>0}$ and $D \in \mathbb{R}$, let $S := \{\nu_1, \nu_2, \ldots, \nu_m\}$ be a multiset of binary strings, each of length s and let $\mathcal{M}$ be the set of medians for S. Then it is

NP-complete to determine if

$$\min_{\mu \in \mathcal{M}} \prod_{i \in [m]} f\left(H(\nu_i, \mu)\right) \leq D. \tag{3.10}$$

We can also state the following corollaries for functions which are strictly concave down.

Corollary 3.14. *Fix a function $f(x) : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ which satisfies the following properties:*

- $\log f(x)$ *is strictly concave down,*
- *the function values can be computed in polynomial time, and*
- *for all but finitely many $n \in \mathbb{Z}$, $n \geq 2$,*

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} \neq 1.$$

For arbitrary $m, s \in \mathbb{Z}^{>0}$ and $D \in \mathbb{R}$, let $S := \{\nu_1, \nu_2, \ldots, \nu_m\}$ be a multiset of binary strings, each of length s. Then it is #P-complete to determine if how many medians μ for S have

$$\prod_{i \in [m]} f\left(H(\nu_i, \mu)\right) \geq D. \tag{3.11}$$

Proof. If the function $f(x)$ has the property that $\log f(x)$ is strictly concave down, then $\log \frac{1}{f(x)}$ is strictly concave up. Therefore by Theorems 3.5 and 3.12 for the function $\frac{1}{f(x)}$, it is #P-hard to determine the number of medians μ which satisfy $\prod_{i \in [m]} \frac{1}{f(H(\nu_i, \mu))} \leq \frac{1}{D}$. This is equivalent to asking for the number of medians μ have $\prod_{i \in [m]} f(H(\nu_i, \mu)) \geq D$. ∎

Corollary 3.15. *Fix a function* $f(x) : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ *which satisfies the following properties:*

- $\log f(x)$ *is strictly concave down,*
- *the function values of* f *can be computed in polynomial time, and*
- *for all but finitely many* $n \in \mathbb{Z}$, $n \geq 2$,

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} \neq 1.$$

For arbitrary $m, s \in \mathbb{Z}^{>0}$ *and* $D \in \mathbb{R}$, *let* $S := \{\nu_1, \nu_2, \ldots, \nu_m\}$ *be a multiset of binary strings, each of length* s *where* $\mathcal{M}$ *is the set of medians for* S. *Then it is NP-complete to determine if*

$$\min_{\mu \in \mathcal{M}} \prod_{i \in [m]} f\left(H(\nu_i, \mu)\right) \geq D. \tag{3.12}$$

3.2 Stochastic Approximations

We have seen several proofs showing that it is hard to calculate many of these quantities. However, we may further ask if any of the quantities can be approximated. One method of approximation is by an FPRAS:

Definition 3.16. *A counting problem* $\#A$ *in* $\#P$ *has an* FPRAS *(fully polynomial randomized approximation scheme) if there is a randomized algorithm such that, for any instance of* $\#A$ *and any* $\epsilon, \delta > 0$, *the algorithm outputs an approximation* $\hat{f}$ *for the solution* f *satisfying*

$$P\left(\frac{f}{1+\epsilon} \leq \hat{f} \leq f(1+\epsilon)\right) \geq 1 - \delta$$

and the algorithm runs in time polynomial in the size of the instance, $\frac{1}{\epsilon}$, *and* $\log \frac{1}{\delta}$.

Before stating our results, we define a couple more complexity classes for decision problems:

Definition 3.17 (Gill (1977))**.** *A decision problem, A, is in the class RP (randomized polynomial time) if there is a probabilistic Turing machine that runs in polynomial time in the size of the input, returns "true" with probability at least* $\frac{1}{2}$ *when the answer for A is true, and returns "false" with probability 1 when the answer for A is false.*

Definition 3.18 (Gill (1977))**.** *A decision problem, A, is in the class BPP (bounded-error probabilistic polynomial time) if there is a probabilistic Turing machine that runs in polynomial time in the size of the input, returns "true" with probability at least* $\frac{2}{3}$ *when the answer for A is true, and returns "false" with probability* $\frac{2}{3}$ *when the answer for A is false.*

One result connecting these classes is the following:

Theorem 3.19 (Papadimitriou (1994))**.** *If the intersection of NP and BPP is non-empty, then RP=NP.*

Note that each result below holds for functions $f(x)$ with $\log f(x)$ strictly concave down. The analogous results for the functions whose logarithm is concave up are still open. Now for our first result regarding sampling of medians for #StarSPSCJ(f).

Theorem 3.20. *Fix a function* $f(x) : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ *which satisfies the following properties:*

- $\log f(x)$ *is strictly concave down,*
- *the function values of* f *can be computed in polynomial time, and*
- *there exists* $\epsilon > 0$ *such that for all but finitely many* $n \in \mathbb{Z}$, $n \geq 2$,

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} < 1 - \epsilon.$$

For arbitrary $m, s \in \mathbb{Z}^{>0}$ *and* $D \in \mathbb{R}$, *let* $S := \{\nu_1, \nu_2, \ldots, \nu_m\}$ *be a multiset of binary strings, each of length s. If there is a rapidly mixing Markov chain with stationary distribution proportional to* $\prod_{i \in [m]} f\left(H(\nu_i, \mu)\right)$, *then RP=NP.*

Proof. Fix a function f as described in the theorem. Because $\log f(x)$ is strictly concave down, $\log(f(x))^{-1}$ is strictly concave up. Set $g(x) := (f(x))^{-1}$.

Now recall the proof of Theorem 3.5 for strictly concave up functions. Take a D3CNF Γ with n variables and create a multiset of binary strings, $\mathcal{D}$. The set of medians for $\mathcal{D}$ is $\mathcal{M} = \{0,1\}^{2n} \times \{0\}^t$. There is a one-to-one correspondence between the medians in the subset $\mathcal{M}' = \{01,10\}^n \times \{0\}^t$ and the truth assignments for Γ. Those medians which correspond to satisfying truth assignments for Γ form the set $\mathcal{M}'_\Gamma$. The multiset $\mathcal{D}$ is constructed so that each $\mu \in \mathcal{M}'_\Gamma$ has

$$\prod_{\eta\in\mathcal{D}} \frac{1}{f(H(\eta,\mu))} = \prod_{\eta\in\mathcal{D}} g(H(\eta,\mu)) = \alpha_{good}\beta_{good}^n\gamma_{good}^k =: h_3$$

and all other medians have

$$\prod_{\eta\in\mathcal{D}} \frac{1}{f(H(\eta,\mu))} = \prod_{\eta\in\mathcal{D}} g(H(\eta,\mu)) > \alpha_{good}\beta_{good}^n\gamma_{bad}\gamma_{good}^{k-1} =: h_2.$$

Equivalently, if $\mu \in \mathcal{M}'_\Gamma$, then

$$\prod_{\eta\in\mathcal{D}} f(H(\eta,\mu)) = \frac{1}{h_3}.$$

Otherwise,

$$\prod_{\eta\in\mathcal{D}} f(H(\eta,\mu)) < \frac{1}{h_2}.$$

Further,

$$\begin{aligned}
\frac{h_2}{h_3} &= \frac{\gamma_{bad}}{\gamma_{good}} \\
&= \frac{g(n-1)[g(n+2)]^3[g(n+3)]^3 g(n+6)}{g(n)[g(n+1)]^3[g(n+4)]^3[g(n+5)]} \\
&= \frac{f(n)[f(n+1)]^3[f(n+4)]^3[f(n+5)]}{f(n-1)[f(n+2)]^3[f(n+3)]^3 f(n+6)} \\
&> \frac{1}{1-\epsilon}
\end{aligned}$$

where the last inequality is a result of the assumption in the theorem statement. As a result

$$\frac{1}{h_2}\left(\frac{1}{1-\epsilon}\right) < \frac{1}{h_3}.$$

Now select an integer r, dependent only on the values of n and ϵ, such that $\left(\frac{1}{1-\epsilon}\right)^r > 2^{2n+2}$. Create a new multiset $\mathcal{D}(r)$ of binary strings such that

$$\mathcal{D}(r) = \underbrace{\mathcal{D} \uplus \ldots \uplus \mathcal{D}}_{r \text{ times}}.$$

The set of medians for $\mathcal{D}(r)$ is the same as the set of medians for $\mathcal{D}$. However this time, for a median $\mu \in \mathcal{M}'_\Gamma$,

$$\prod_{\eta \in \mathcal{D}(r)} f(H(\eta, \mu)) = \left(\frac{1}{h_3}\right)^r.$$

Otherwise, if $\mu \in \mathcal{M} \setminus \mathcal{M}'_\Gamma$,

$$\prod_{\eta \in \mathcal{D}(r)} f(H(\eta, \mu)) < \left(\frac{1}{h_2}\right)^r.$$

By the choice of r,

$$\left(\frac{1}{h_2}\right)^r 2^{2n} < \left(\frac{1}{h_2}\right)^r 2^{2n+2} < \left(\frac{1}{h_2}\left(\frac{1}{1-\epsilon}\right)\right)^r < \left(\frac{1}{h_3}\right)^r.$$

Since $\mathcal{M} = \{0,1\}^{2n} \times \{0\}^t$, $|\mathcal{M}| = 2^{2n}$ and the above inequality shows that for each $\mu_0 \in \mathcal{M}'_\Gamma$,

$$\prod_{\eta \in \mathcal{D}} f(H(\eta, \mu_0)) > \sum_{\mu \in \mathcal{M} \setminus \mathcal{M}'_\Gamma} \prod_{\eta \in \mathcal{D}} f(H(\eta, \mu)).$$

Further,

$$\prod_{\eta \in \mathcal{D}} f(H(\eta, \mu_0)) > \frac{1}{2} \sum_{\mu \in \mathcal{M}} \prod_{\eta \in \mathcal{D}} f(H(\eta, \mu)).$$

Now suppose that we had a rapidly mixing Markov chain for this instance of #StarSPSCJ(f), as stated in the theorem. From the calculations above, it must sample medians which correspond to satisfying truth assignments for Γ with probability at least $\frac{1}{2}$. This is precisely an RP for D3SAT. However, this immediately implies RP=NP because D3SAT is NP-complete. ∎

The following theorem gives the same result as the last one for different functions f. In particular, it switches the inequality that f is required to satisfy.

Theorem 3.21. *Fix a function* $f(x) : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ *which satisfies the following properties:*

- $\log f(x)$ *is strictly concave down,*
- *the function values of* f *can be computed in polynomial time, and*
- *there exists* $\epsilon > 0$ *such that for all but finitely many* $n \in \mathbb{Z}$, $n \geq 2$,

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} > 1 + \epsilon.$$

For arbitrary $m, s \in \mathbb{Z}^{>0}$ *and* $D \in \mathbb{R}$, *let* $S := \{\nu_1, \nu_2, \ldots, \nu_m\}$ *be a multiset of binary strings, each of length* s. *If there is a rapidly mixing Markov chain with distribution proportional to* $\prod_{i \in [m]} f\left(H(\nu_i, \mu)\right)$, *then RP=NP.*

Proof. The proof for this theorem follows the same line of reasoning as the proof for Theorem 3.20. However, it makes use of details in Theorem 3.12 rather than Theorem 3.5. ∎

When f is a function with $\log f(x)$ is concave down, we examine the possibility of an FPRAS (Definition 3.16) for #StarSPSCJ(f) .

Theorem 3.22. *Fix a function* $f(x) : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ *for which:*

- $\log f(x)$ *is strictly concave down,*
- *the function values of* f *can be computed in polynomial time, and*
- *there exists* $\epsilon > 0$ *such that for all but finitely many* $n \in \mathbb{Z}$, $n \geq 2$,

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} < 1 - \epsilon.$$

For arbitrary $m, s \in \mathbb{Z}^{>0}$ *and* $D \in \mathbb{R}$, *let* $S := \{\nu_1, \nu_2, \ldots, \nu_m\}$ *be a multiset of binary strings, each of length* s. *If there is an FPRAS for calculating*

$$\sum_{\mu \in \mathcal{M}} \prod_{i \in [m]} f(H(\nu_i, \mu)),$$

then RP=NP.

Proof. Let r be an integer so that $\left(\frac{1}{1-\epsilon}\right)^r > 2^{2n+2}$. In the proof of Theorem 3.20, we created a new multiset of genomes $\mathcal{D}(r)$. The set of medians $\mathcal{M}$ for $\mathcal{D}(r)$ is precisely $\{0,1\}^{2n} \times \{0\}^t$ and each median $\mu \in \mathcal{M}'_\Gamma$ which corresponds to a satisfying truth assignment for Γ has

$$\prod_{\eta \in \mathcal{D}(r)} f(H(\eta, \mu)) = \left(\frac{1}{h_3}\right)^r.$$

All other medians have

$$\prod_{\eta \in \mathcal{D}(r)} f(H(\eta, \mu)) < \left(\frac{1}{h_2}\right)^r.$$

Therefore, if Γ has no satisfying assignments,

$$\sum_{\mu \in \mathcal{M}} \prod_{\eta \in \mathcal{D}(r)} f(H(\eta, \mu)) < 2^{2n} \left(\frac{1}{h_2}\right)^r.$$

If there is a satisfying assignment for Γ, then

$$\sum_{\mu \in \mathcal{M}} \prod_{\eta \in \mathcal{D}(r)} f(H(\eta, \mu)) \geq \left(\frac{1}{h_3}\right)^r.$$

By the choice of r, we have the following inequality to relate the two quantities:

$$\left(\frac{1}{h_2}\right)^r 2^{2n+2} < \left(\frac{1}{h_3}\right)^r.$$

Now suppose that there is an FPRAS for $T := \sum_{\mu \in \mathcal{M}} \prod_{\eta \in \mathcal{D}(r)} f(H(\eta, \mu))$. In other words, for any $\epsilon, \delta > 0$, there is a randomized algorithm as described in Definition 3.16 which outputs a quantity $\hat{T}$ such that

$$P\left(\frac{T}{1+\epsilon} \leq \hat{T} \leq T(1+\epsilon)\right) \geq 1 - \delta.$$

Consider the case when $\delta = \frac{1}{3}$ and $\epsilon = 1$. Therefore,

$$P\left(\frac{1}{2}T \leq \hat{T} \leq 2T\right) \geq \frac{2}{3}.$$

Therefore, if Γ can be satisfied, then $T \geq \left(\frac{1}{h_3}\right)^r$ and the probability that $\hat{T}$ is at least $\frac{1}{2}T = \frac{1}{2}\left(\frac{1}{h_3}\right)^r > 2^{2n+1}\left(\frac{1}{h_2}\right)^r$ is $\frac{2}{3}$. On the other hand, if Γ cannot be satisfied, then

$T < 2^{2n}\left(\frac{1}{h_2}\right)^r$ and the probability that $\hat{T}$ is at most $2T = 2^{2n+1}\left(\frac{1}{h_2}\right)^r$ is $\frac{2}{3}$. Therefore, we have a BPP algorithm (Definition 3.18) for 3SAT.

Because 3SAT is NP-complete, Papadimitriou's Theorem 3.19 implies RP=NP. ∎

A similar result holds for functions $f(x)$ which satisfy the opposite inequality. We do not give a proof as it follows the reasoning in the proof of Theorem 3.22.

Theorem 3.23. *Fix a function $f(x) : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ for which:*

- $\log f(x)$ *is strictly concave down,*
- *the function values of f can be computed in polynomial time, and*
- *there exists $\epsilon > 0$ such that for all but finitely many $n \in \mathbb{Z}$, $n \geq 2$,*

$$\frac{f(n-2)[f(n+1)]^3[f(n+2)]^3 f(n+5)}{f(n-1)[f(n)]^3[f(n+3)]^3 f(n+4)} > 1+\epsilon.$$

For arbitrary $m, s \in \mathbb{Z}^{>0}$ and $D \in \mathbb{R}$, let $S := \{\nu_1, \nu_2, \ldots, \nu_m\}$ be a multiset of binary strings, each of length s. If there is an FPRAS for calculating

$$\sum_{\mu \in \mathcal{M}} \prod_{i \in [m]} f(H(\nu_i, \mu)),$$

then RP=NP.

Chapter 4

Result for Binary Phylogenetic Trees

In this chapter, we return our attention to counting most parsimonious SCJ scenarios, but this time our phylogenetic trees are binary trees. We use the following definition.

Definition 4.1. *A tree is a binary tree if it is rooted and every non-leaf vertex has exactly two children.*

Recall the statement of #BinSPSCJ:

Definition 1.25. *Given arbitrary integer $m \geq 2$, let T be a binary tree with m leaves. Let $B = \{\nu_i\}_{i=1}^{m}$ be an arbitrary multiset of binary strings and a surjective function $\varphi : L(T) \to B$. Define F to be the set of most parsimonious labelings φ' which extend φ to $V(T)$. Determine the value of*

$$\sum_{\varphi' \in F} \prod_{uv \in E(T)} H(\varphi'(u), \varphi'(v))!.$$

The main result of Section 4.2 is the theorem which states #BinSPSCJ is in #P-complete. In the first section, we develop several tools and algorithms which lay the foundation for our main theorem.

4.1 Algorithms for finding most parsimonious labelings

Let $\Gamma = c_1 \wedge c_2 \wedge \ldots \wedge c_k$ be a D3CNF with variables $\{v_1, v_2, \ldots, v_n\}$. Select new variables $\{w_1, w_2, \ldots, w_n\}$ which do not occur in Γ. For each $i \in [n]$, understanding subscript $n+1$ as 1, define the following D3CNF,

$$\Phi_i := (v_i \vee w_i \vee v_{i+1}) \wedge (v_i \vee w_i \vee \overline{v_{i+1}}) \wedge (\overline{v_i} \vee \overline{w_i} \vee v_{i+1}) \wedge (\overline{v_i} \vee \overline{w_i} \vee \overline{v_{i+1}}). \quad (4.1)$$

Observe that Φ_i is equivalent to the "exclusive or" $(v_i \vee w_i) \wedge (\overline{v_i} \vee \overline{w_i})$. Define

$$\Psi(\Gamma) := \Gamma \wedge \bigwedge_{i=1}^{n} \Phi_i.$$

Necessarily, if Γ is a D3CNF then so is $\Psi(\Gamma)$.

Lemma 4.2. *For Γ, an arbitrary D3CNF, it is #P-complete to determine the number of satisfying truth assignments for $\Psi(\Gamma)$.*

Proof. We have already shown in Lemma 1.20 that #D3SAT is in #P-complete. So to prove this result, we will show that the satisfying truth assignments for Γ and for $\Psi(\Gamma)$ are in one-to-one correspondence.

Any truth assignment which satisfies $\Psi(\Gamma)$, when restricted to $\{v_1, v_2, \ldots, v_n\}$ will necessarily satisfy Γ.

For the other direction, recall that Φ_i is equivalent to the "exclusive or" for v_i and w_i. Therefore, given a satisfying truth assignment for Γ, we can create a unique satisfying truth assignment for $\Psi(\Gamma)$ by assigning to each w_i the opposite value of v_i. ∎

Next we provide two different algorithms for creating most parsimonious labelings given a rooted binary tree and a leaf-labeling $\varphi : L(T) \to \{0,1\}^s$. If we restrict $\varphi(\ell)$ to a single coordinate c for every leaf ℓ, we obtain a labeling $\varphi_c : L(T) \to \{0,1\}$. Each algorithm presented below will consider leaf labels from the set $\{0,1\}$ and output a most parsimonious labeling $\varphi'_c : V(T) \to \{0,1\}$. Obtaining a most parsimonious labeling for each coordinate in this way, we combine these labelings to create a most parsimonious labeling $\varphi' : V(T) \to \{0,1\}^s$ for T and the original leaf-labeling φ.

Let T be a binary tree with root ρ and let $\varphi : L(T) \to \{0,1\}$ be a labeling for the leaves. Let $\varphi' : V(T) \to \{0,1\}$ be a most parsimonious labeling (Definition 1.4) which extends φ. Because each vertex is labeled with a single bit, $H(\varphi'(u), \varphi'(v)) \in \{0,1\}$. By definition, the most parsimonious labeling φ' mini-

mizes the sum $\sum_{uv \in E(T)} H(\varphi'(u), \varphi'(v))$. Consequently, φ' must minimize the number of edges uv such that $\varphi'(u) \neq \varphi'(v)$.

First, we have Fitch's algorithm to find most parsimonious labelings.

Fitch's Algorithm (Fitch (1971))**.** *Let T be a binary tree with root ρ and leaf-labeling $\varphi : L(T) \to \{0, 1\}$. The following algorithm, completed in two parts, will find a most parsimonious labeling $\varphi' : V(T) \to \{0, 1\}$ which extends φ.*

Part 1: Define a function B on the vertices of T as follows: For each leaf ℓ, set $B(\ell) := \{\varphi(\ell)\}$. Extend this assignment to all vertices of T by the following rule: For a vertex u with children v_1, v_2 such that $B(v_1)$ and $B(v_2)$ have been defined, set

$$B(u) := \begin{cases} B(v_1) \cap B(v_2) & \text{if } B(v_1) \cap B(v_2) \neq \varnothing, \\ B(v_1) \cup B(v_2) & \text{otherwise.} \end{cases} \tag{4.2}$$

Part 2: Select a single element $\alpha \in B(\rho)$. Define a function φ' on the vertices of T as follows: Set $\varphi'(\rho) := \alpha$. Extend φ' to $V(T)$ by the following rule: If v is a child of u and $\varphi'(u)$ is defined, then

$$\varphi'(v) := \begin{cases} \varphi'(u) & \text{if } \varphi'(u) \subseteq B(v), \\ 1 - \varphi'(u) & \text{if } \varphi'(u) \notin B(v). \end{cases} \tag{4.3}$$

The resulting φ' is a most parsimonious labeling extending φ and is called a Fitch solution.

While Fitch solutions are most parsimonious labelings, there are cases when Fitch's algorithm finds some of the most parsimonious labelings but not all of them. However, Sankoff's algorithm, described below, will produce all most parsimonious labelings (Erdős and Székely 1994).

Sankoff's Algorithm (Erdős and Székely (1994); Sankoff and Rousseau (1975))**.** *Let T be a binary tree with root ρ and leaf labeling $\varphi : L(T) \to \{0, 1\}$. This algorithm is completed in two steps.*

Part 1: Define functions s_0 and s_1 on the vertices of T as follows: For each leaf ℓ,

$$s_0(\ell) := \begin{cases} 0 & \text{if } \varphi(\ell) = 0, \\ \infty & \text{otherwise.} \end{cases} \tag{4.4}$$

$$s_1(\ell) := \begin{cases} 0 & \text{if } \varphi(\ell) = 1, \\ \infty & \text{otherwise.} \end{cases}$$

Extend these functions recursively to all vertices by the following: If v_0 and v_1 are children of u and $s_i(v_j)$ has been defined for all $i, j \in \{0, 1\}$, then

$$s_0(u) := \min\{s_0(v_0), s_1(v_0) + 1\} + \min\{s_0(v_1), s_1(v_1) + 1\}, \tag{4.5}$$

$$s_1(u) := \min\{s_0(v_0) + 1, s_1(v_0)\} + \min\{s_0(v_1) + 1, s_1(v_1)\}. \tag{4.6}$$

Note: For any $v \in V(T)$, $s_i(v)$ counts the minimum number of edges, within the subtree containing v and its descendants, that will witness a change if a most parsimonious solution labeled v with the value i. A leaf will have $s_0(\ell) = \infty$ (or $s_1(\ell) = \infty$ if it is impossible for a most parsimonious labeling to label ℓ with a 0 (1), because most parsimonious labelings must agree with the original leaf label.

Part 2: For each $v \in V(T)$, select $\alpha_v \in \{0, 1\}$. Define the function φ' on the vertices of T as follows: For root ρ, define

$$\varphi'(\rho) := \begin{cases} 0 & \text{if } s_0(\rho) < s_1(\rho), \\ \alpha_\rho & \text{if } s_0(\rho) = s_1(\rho), \\ 1 & \text{if } s_0(\rho) > s_1(\rho). \end{cases}$$

Extend φ' to $V(T)$ by the following rule: If v is a child of u and $\varphi'(u)$ is defined, then define $\varphi'(v)$ as follows: If $\varphi'(u) = 0$, then

$$\varphi'(v) := \begin{cases} 0 & \text{if } s_0(v) < s_1(v) + 1, \\ \alpha_v & \text{if } s_0(v) = s_1(v) + 1, \\ 1 & \text{if } s_0(v) > s_1(v) + 1. \end{cases} \tag{4.7}$$

If $\varphi'(u) = 1$, then

$$\varphi'(v) := \begin{cases} 1 & \text{if } s_1(v) < s_0(v) + 1, \\ \alpha_v & \text{if } s_1(v) = s_0(v) + 1, \\ 0 & \text{if } s_1(v) > s_0(v) + 1. \end{cases} \tag{4.8}$$

The resulting φ' is a most parsimonious labeling for T extending φ and is called a Sankoff solution.

The following lemma draws a connection between the solutions found from each algorithm.

Lemma 4.3. *Let T be a binary tree with leaf-labeling $\varphi : L(T) \to \{0, 1\}$. Suppose that, for each $u, v \in V(T)$ with v a child of u, the function B in Fitch's algorithm satisfies*

$$B(v) = \{0, 1\} \Rightarrow B(u) = \{0, 1\}. \tag{4.9}$$

Then for T and φ, all Sankoff solutions are Fitch solutions. In other words, Fitch's algorithm finds all most parsimonious labelings.

In order to prove Lemma 4.3, we first establish a series of claims (4.4 through 4.7) under the assumptions of Lemma 4.3.

Claim 4.4. *For any non-leaf vertex v, if $B(v) = \{x\}$ for some $x \in \{0, 1\}$, in Sankoff's algorithm $s_0(v) = 0$ and $s_1(v) = 2$.*

Proof. The proof proceeds by reverse induction on the distance from the root. For the base case, we consider those vertices whose children are both leaves. Let v be such a vertex with children v_ℓ and v_r. By symmetry of the argument, assume $B(v) = \{0\}$. Then $B(v_\ell) = B(v_r) = \{0\}$ which only happens for leaves if $\varphi(v_\ell) = \varphi(v_r) = 0$. By (4.4), $s_0(v_\ell) = s_0(v_r) = 0$ and $s_1(v_\ell) = s_1(v_r) = \infty$. As desired, (4.5) implies $s_0(v) = 0$ and (4.6) implies $s_1(v) = 2$.

For the inductive hypothesis, assume that each vertex v of distance at least $d \geq 1$ from the root has either $s_0(v) = 0$ and $s_1(v) = 2$ or $s_0(v) = 2$ and $s_1(v) = 0$. Let u be a vertex of distance $d - 1$ from the root. Again, we assume $B(u) = \{0\}$ as the argument for the case when $B(u) = \{1\}$ is very similar. This vertex has two children, u_ℓ and u_r. There are three cases to consider.

(1) If u_ℓ and u_r are leaves, then the argument in the base case gives $s_0(u) = 0$ and $s_1(u) = 2$ as desired.

(2) If u_ℓ is a leaf and u_r is not a leaf, then $s_0(u_r) = 0$ and $s_1(u_r) = \infty$ and, by the inductive hypothesis $s_0(u_\ell) = 0$ and $s_1(u_\ell) = 2$. Therefore (4.5) implies $s_0(u) = 0$ and (4.6) implies $s_1(u) = 2$.

(3) If u_ℓ and u_r are not leaves, then by the inductive hypothesis, $s_0(u_\ell) = s_0(u_r) = 0$ and $s_1(u_\ell) = s_1(u_r) = 2$. Again, (4.5) implies $s_0(u) = 0$ and (4.6) implies $s_1(u) = 2$.

This complete the proof of the claim. ∎

Claim 4.5. *For any vertex v with $B(v) = \{0, 1\}$ from Fitch's algorithm, we will have $s_0(v) = s_1(v)$ in Sankoff's algorithm.*

Proof. This claim is also proven by induction on distance from the root where the base case examines those vertices with greatest distance from the root.

For the base case, let v be a vertex with $B(v) = \{0, 1\}$ and none of its descendants u have $B(u) = \{0, 1\}$. For children v_ℓ and v_r of v we may assume $B(v_\ell) = \{0\}$ and $B(v_r) = \{1\}$ by (4.2). By Claim 4.4,

$$s_0(v_\ell) = s_1(v_r) = 0 \text{ and } s_0(v_r) = s_1(v_\ell) = 2.$$

Therefore $s_0(v) = s_1(v) = 1$.

For the inductive hypothesis, suppose all vertices u with $B(u) = \{0, 1\}$ of distance at least $d \geq 1$ from the root have $s_0(u_\ell) = s_1(u_r)$. Let v be a vertex at distance $d - 1$ from the root with $B(v) = \{0, 1\}$. There are three cases to consider:

(1) If v has a child v_ℓ with $B(v_\ell) = \{0\}$, then by (4.2) the other child v_r must have $B(v_r) = \{1\}$ and we can use the argument in the base case to see $s_0(v_\ell) = s_1(v_r)$.

(2) If v has a child v_ℓ with $B(v_\ell) = \{1\}$, then by (4.2), v must have another child v_r with $B(v_r) = \{0\}$. This puts us back in case 1.

(3) If v has a child v_ℓ with $B(v_\ell) = \{0, 1\}$, then by (4.2), v must have another child v_r with $B(v_r) = \{0, 1\}$. By the inductive hypothesis, $s_0(v_\ell) = s_1(v_\ell)$ and $s_0(v_r) = s_1(v_r)$. By (4.5) and (4.6), $s_0(v) = s_1(v)$.

This completes the proof of the claim. ∎

Claim 4.6. *For any non-leaf vertex v with $B(v) = \{i\}$ ($i \in \{0, 1\}$), both Fitch's algorithm and Sankoff's algorithm will define $\varphi'(v) = i$.*

Proof. In Fitch's algorithm, this is an immediate consequence of (4.3).

Now consider Sankoff's algorithm. If $B(v) = \{0\}$ then, by Claim 4.4, $s_0(v) = 0$ and $s_1(v) = 2$. Observe $s_0(v) < s_1(v) + 1$ and $s_1(v) > s_0(v) + 1$. Therefore $\varphi'(v) = 0$ by (4.7) and (4.8). ∎

Claim 4.7. *Suppose $B(\rho) = \{0, 1\}$. For any vertex v with $B(v) = \{0, 1\}$, if both algorithms set $\varphi'(\rho) := 0$, then both Fitch's algorithm and Sankoff's algorithm will set $\varphi'(v) = 0$. Likewise, if $\varphi'(\rho) = 1$, then both algorithms will set $\varphi'(v) = 1$.*

Proof. For any vertex v with $B(v) = \{0, 1\}$, there is a path $\rho = u_0, u_1, \ldots, u_{t-1}, u_t = v$ of vertices such that $B(u_i) = \{0, 1\}$ for each $i \in [t]$. It suffices to show that, in both algorithms, if $\varphi'(u_i) = 0$ for some $0 \leq i < t$, then $\varphi'(u_{i+1}) = 0$.

In Part 2 of Fitch's algorithm, if $\varphi'(u_i) = 0$ and $B(u_{i+1}) = \{0, 1\}$, then (4.2) implies $\varphi'(u_{i+1}) = 0$.

In Sankoff's algorithm, if $\varphi'(u_i) = 0$ and $B(u_{i+1}) = \{0, 1\}$, then by Claim 4.7, $s_0(u_{i+1}) = s_1(u_{i+1})$. Thus $s_0(u_{i+1}) < s_1(u_{i+1}) + 1$. Since $\varphi'(u_i) = 0$, (4.7) implies $\varphi'(u_{i+1}) = 0$.

A similar argument can be used to show that if $\varphi'(\rho) = 1$, then $\varphi'(v) = 1$. Therefore Fitch's algorithm and Sankoff's algorithm will agreed on $\varphi'(v)$ if they agree on $\varphi'(\rho)$. ∎

Proof of Lemma 4.3. In each algorithm, once $\varphi(\rho)$ has been set, the algorithm deterministically outputs a most parsimonious labeling of all vertices. Therefore, it suffices to prove that both algorithms have the same choices for labeling the root and both algorithms output the same most parsimonious labeling for the same choice for $\varphi'(\rho)$.

If $B(\rho) = \{0\}$ or $B(\rho) = \{1\}$, then there is only one choice in Fitch's algorithm for $\varphi'(\rho)$. By Claim 4.4, Sankoff's algorithm has the same determined value for $\varphi'(\rho)$. Further, all vertices $v \in V(T)$ will have either $B(v) = \{0\}$ or $B(v) = \{1\}$ by condition (4.9) and Claim 4.6 completes the proof.

If $B(\rho) = \{0, 1\}$, then in Fitch's algorithm, there are two choices for $\varphi'(\rho)$. By Claim 4.5, $s_0(\rho) = s_1(\rho)$ in Sankoff's algorithm, which means there are also two choices for $\varphi'(\rho)$. Claim 4.7 implies that if we make the same choice for the root, both algorithms give the same most parsimonious labeling φ'.

Sankoff's algorithm is guaranteed to find all most parsimonious labelings and the most parsimonious labelings from Fitch's algorithm coincide with those from Sankoff's algorithm, this implies that Fitch's algorithm finds all most parsimonious labelings. ∎

As mentioned earlier, these algorithms are designed for a tree T with leaf-labeling $\varphi : L(T) \to \{0,1\}$. However, given a tree T with leaf-labeling $\phi : L(T) \to \{0,1\}^N$, we can restrict all strings to a single coordinate and run one of the above algorithms to find a most parsimonious labelings for $V(T)$ in that coordinate. Repeat this for each coordinate. The most parsimonious labelings found for each coordinate can then be combined into a most parsimonious labeling of $V(T)$ that extends ϕ.

4.2 Complexity result for #BinSPSCJ

Here is our main result on binary trees.

Theorem 4.8. *#BinSPSCJ is #P-complete.*

Proof. In Lemma 1.23 we saw that #SPSCJ is in #P. To prove #SPSCJ is in #P-hard, we provide a polynomial reduction from #D3SAT.

Fix a D3CNF, $\Gamma = \bigwedge_{i\in[k]} c_i$ with k clauses and n variables. Let

$$\Psi(\Gamma) := \bigwedge_{i\in[k]} c_i \wedge \bigwedge_{i\in[n]} \Phi_i$$

with $2n$ variables, $\{v_1, v_2, \ldots v_n, w_1, w_2, \ldots, w_n\}$, where each clause c_i has three distinct literals from $\{v_i, \overline{v_i} : i \in [n]\}$, and Φ_i is the D3CNF in (4.1) which guarantees that, for each $i \in [n]$, v_i and w_i have different truth values. By Lemma 4.2, Γ and $\Psi(\Gamma)$ have the same satisfying truth assignments. We will construct a binary tree $\mathcal{B}$ and define a labeling φ of its leaves such that the number of satisfying truth assignment for $\Psi(\Gamma)$ is directly computable from the number of most parsimonious SCJ scenarios on $\mathcal{B}$.

Each Φ_i has 4 clauses, so $\Psi(\Gamma)$ has $k+4n$ clauses. Assign the names $c_{k+1}, \ldots, c_{k+4n}$ to the $4n$ clauses of $\bigwedge_{i\in[n]} \Phi_i$. Then

$$\Psi(\Gamma) = \bigwedge_{i\in[k+4n]} c_i.$$

For each $i \in [k+4n]$, we define a binary tree $\mathcal{B}_i$ which encodes clause c_i. The final binary tree $\mathcal{B}$ will join $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_{k+4n}$ by a comb. For $t = 148(16n^2+8kn)(k+4n)$, the leaf-labeling $\varphi : L(\mathcal{B}) \to \{0,1\}^{2n+t}$, will assign a binary string with coordinates $(x_1, y_1, \ldots, x_n, y_n, e_1, \ldots, e_t)$ to each leaf. The x_i coordinates will correspond to the v_i variables and the y_i coordinates will correspond to the w_i variables of $\Psi(\Gamma)$. The e_i coordinates will be for additional ones, used in a manner similar to the additional ones in the previous two chapters for star trees.

In this chapter, we denote the left child of a non-leaf vertex v by v_ℓ and the right child by v_r. The height of a vertex is its graph distance from the root. The construction of $\mathcal{B}_i$ with its leaf labeling φ will come in Definition 4.12, but first we need some preliminary definitions.

For any clause $c_i = v_\alpha \vee v_\beta \vee v_\gamma$ which is the disjunction of 3 distinct literals, Miklós, Kiss, and Tannier (2014) defined a *unit subtree*, $\mathcal{U}_i$, with 248 leaves. They also defined a leaf-labeling $\hat{\varphi} : L(\mathcal{U}_i) \to \{0,1\}^{151}$ where the binary strings in the range have coordinates $(\hat{x}_\alpha, \hat{x}_\beta, \hat{x}_\gamma, \hat{e}_1, \hat{e}_2, \ldots, \hat{e}_{148})$. The first three coordinates correspond to the variables in c_i and the remaining 148 coordinates are for additional ones. This unit subtree has some useful properties which will be discussed after Definition 4.12.

For each $i \in [k+4n]$, let $\mathcal{U}_i$ be the unit subtree for clause c_i. If $i \leq k$ where c_i relates $v_\alpha, v_\beta, v_\gamma$, then $\mathcal{U}_i$ will have leaf labels with coordinates $\{x_\alpha, x_\beta, x_\gamma\}$ and 148 coordinates for additional ones. If $i > k$ where c_i relates variables $v_\alpha, w_\alpha, v_{\alpha+1}$, $\mathcal{U}_i$ will have leaf labels with coordinates $\{x_\alpha, y_\alpha, x_{\alpha+1}\}$ and 148 coordinates for additional ones.

Definition 4.9. *The tree $\mathcal{T}_i$ in Step 5 of Definition 4.11 is a comb joining $16n^2+8kn$ copies of $\mathcal{U}_i$, as in Figure 4.1.*

Definition 4.10. *For three literals a, b, c, we define $S(a,b,c)$ to be the complete binary tree of height 3 with root ρ with the vertices labeled with equations as follows:*

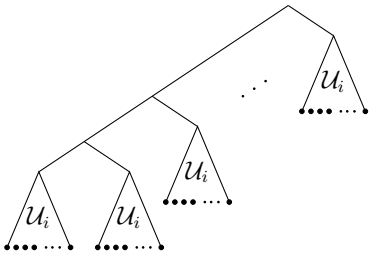


Figure 4.1 Comb connecting
$16n^2 + 8kn$ copies of $\mathcal{U}_i$ to create $\mathcal{T}_i$

1. *Assign the label "$a = 0$" to vertex ρ_ℓ and "$a = 1$" to ρ_r.*

2. *For each vertex u of height 1, assign the label "$b = 0$" to u_ℓ and "$b = 1$" to u_r.*

3. *For each vertex v of height 2, assign the label "$c = 0$" to v_ℓ and "$c = 1$" to v_r.*

This tree is pictured on the right in Figure 4.2. We will use the representation on the left in place of $S(a, b, c)$ in future figures.

Next we construct $\hat{\mathcal{B}}_i$ which will have the same tree structure as $\mathcal{B}_i$. However, $\hat{\mathcal{B}}_i$ will have all of its vertices labeled with equations while $\mathcal{B}_i$ will only have leaf labels which are binary strings. The leaf labeling of $\mathcal{B}_i$ will be induced by the vertex labels of $\hat{\mathcal{B}}_i$. Each leaf will essentially inherit the labels of its ancestors.

Definition 4.11. *Fix $i \in [k + 4n]$. Construct $\hat{\mathcal{B}}_i$, a binary tree with vertex labels, as follows.*

A. *If $i \in [k]$, then say clause c_i has variables $v_\alpha, v_\beta, v_\gamma$. The construction of $\hat{\mathcal{B}}_i$ described below is drawn in Figure 4.2.*

   a) *Draw a vertex ρ^i with two children, ρ^i_ℓ and ρ^i_r.*

   b) *Label vertex ρ^i_ℓ with the equations "$x_j = y_j = 0$" for each $j \in [n] \setminus \{\alpha, \beta, \gamma\}$. Label ρ^i_r with "$x_j = y_j = 1$" for all $j \in [n] \setminus \{\alpha, \beta, \gamma\}$.*

   c) *From each of ρ^i_ℓ and ρ^i_r, hang a copy of $S(y_\alpha, y_\beta, y_\gamma)$.*

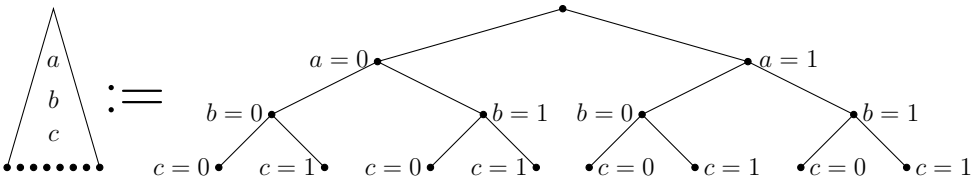


Figure 4.2 The labeled binary tree on the right is $S(a, b, c)$. The representation on the left will be used in place of $S(a, b, c)$ in future figures.

d) From each leaf of each copy of $S(y_\alpha, y_\beta, y_\gamma)$, hang a copy of $S(x_\alpha, x_\beta, x_\gamma)$.

e) Delete the left-most copy of $S(x_\alpha, x_\beta, x_\gamma)$, the one which hangs below the vertices with labels "$y_\alpha = 0$," "$y_\beta = 0$," "$y_\gamma = 0$," and with ancestor ρ^i_ℓ, and replace it with a copy of $\mathcal{T}_i$ from Definition 4.9.

B. If $i \in \{k+1, \ldots, k+4n\}$, then clause c_i relates variables $v_\alpha, w_\alpha, v_{\alpha+1}$. The construction of $\hat{\mathcal{B}}_i$ described below requires only a change of variables from the previous construction.

a) Draw a vertex ρ^i with two children, ρ^i_ℓ and ρ^i_r.

b) Label ρ^i_ℓ with "$x_j = y_j = 0$" for all $j \in [n] \setminus \{\alpha, \alpha+1, \alpha+2\}$. Label ρ^i_r with the system of equations "$x_j = y_j = 1$" for each $j \in [n] \setminus \{\alpha, \alpha+1, \alpha+2\}$.

c) From each of ρ^i_ℓ and ρ^i_r, hang a copy of $S(y_{\alpha+1}, x_{\alpha+2}, y_{\alpha+2})$.

d) Hang a copy of $S(x_\alpha, y_\alpha, x_{\alpha+1})$ from each leaf of each and every copy of $S(y_{\alpha+1}, x_{\alpha+2}, y_{\alpha+2})$.

e) Delete the left-most copy of $S(x_\alpha, y_\alpha, x_{\alpha+1})$ and replace it with a copy of $\mathcal{T}_i$ from Definition 4.9.

Recall $\mathcal{B}$ is a comb connecting binary trees $\mathcal{B}_i$. The binary tree $\mathcal{B}$ has a leaf labeling $\varphi : L(\mathcal{B}) \to \{0,1\}^{2n+t}$ where

$$t = 148(16n^2 + 8kn)(k + 4n).$$

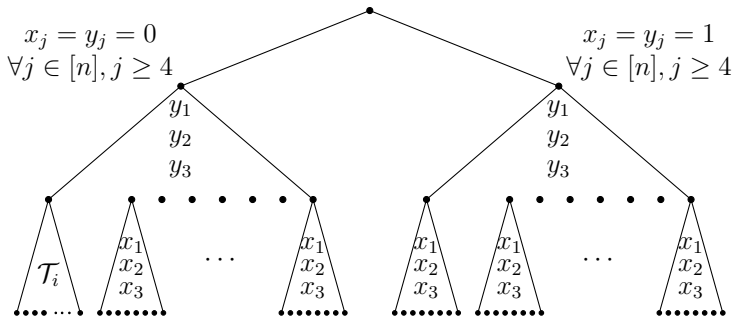


Figure 4.3 The binary tree $\hat{\mathcal{B}}_i$, for $i \in [k]$ created for clause $c_i = x_1 \vee x_2 \vee x_3$.

Each leaf label will have coordinates $(x_1, y_1, \ldots, x_n, y_n, e_1, \ldots, e_t)$. In the next definition, we define $\mathcal{B}_i$ and values of φ on the leaves of $\mathcal{B}_i$.

Definition 4.12. *For each $i \in [k + 4n]$, the binary tree $\mathcal{B}_i$ will have the same tree structure as $\hat{\mathcal{B}}_i$. We only need to explain the labeling $\varphi : L(\mathcal{B}_i) \to \{0,1\}^{2n+t}$.*

Partition $[t]$ into classes E_{ij} with $|E_{ij}| = 148$ for each $i \in [k + 4n]$ and each $j \in [16n^2 + 8kn]$. Identify the set E_{ij} with the j^{th} copy of $\mathcal{U}_i$ in $\mathcal{T}_i$. Here we define $\varphi(\ell)$ for each $\ell \in L(\mathcal{B}_i)$.

There are two cases:

- *If leaf ℓ is not in subtree $\mathcal{T}_i$, then $\varphi(\ell)[e_s] = 0$ for all $s \in [t]$. The value of each $\varphi(\ell)[x_w]$ and $\varphi(\ell)[y_w]$ for $w \in [n]$ is inherited from the labels of the ancestors of ℓ as they appeared in $\hat{\mathcal{B}}_i$.*

- *If leaf ℓ is in the subtree $\mathcal{T}_i$ within $\hat{\mathcal{B}}_i$, then it is a leaf within the j^{th} copy of unit subtree $\mathcal{U}_i$ for some $j \in [16n^2 + 8kn]$. Recall $\hat{\varphi}(\ell) \in \{0,1\}^{151}$ Identify the coordinates $\{\hat{e}_1, \hat{e}_2, \ldots, \hat{e}_{148}\}$ with the indices in the 148 coordinates of E_{ij} in any order. If $\hat{e}_s$ corresponds to coordinate e_r for $r \in E_{ij}$, then we require $\varphi(\ell)[e_r] = \hat{\varphi}(\ell)[\hat{e}_s]$. For each coordinate z which corresponds to a variable in c_i, we also require that $\varphi(\ell)[z] = \hat{\varphi}(\ell)[\hat{z}]$. Set $\varphi(\ell)[e_s] = 0$ for $s \notin E_{ij}$. All other*

coordinates of $\varphi(\ell)$ will take the value 0 (the value inherited from the labeling of their ancestors in $\hat{\mathcal{B}}_i$.

Define $\varphi_{x_j} : L(\mathcal{B}) \to \{0,1\}$ so that $\varphi_{x_j}(\ell) = \varphi(\ell)[x_j]$. Define φ_{y_j} and φ_{e_j} similarly. We want to examine the Fitch solutions on $\mathcal{B}$ for each φ_{x_j}, φ_{y_i}, and φ_{e_j}. We will first prove that the conditions of Lemma 4.3 hold and thus Fitch's algorithm find all most parsimonious labelings for φ on $\mathcal{B}$.

We first explore the Fitch solutions for φ_{e_j}, $j \in [t]$.

Fact 4.13. *Fix $j \in [t]$. There is only one $\ell \in L(\mathcal{B})$ with $\varphi_{e_j}(\ell) = 1$. After running Part 1 of Fitch's algorithm, $B(\ell) = \{1\}$, the parent v of ℓ has $B(v) = \{0,1\}$, and $B(u) = \{0\}$ for all other vertices. Consequently, Part 2 of Fitch's algorithm will output a most parsimonious labeling φ'_{e_j} such that $\varphi'_{e_j}(\ell) = 1$ and for all other vertices $u \in V(\mathcal{B})$, $\varphi'_{e_j}(u) = 0$.*

Proof. These values of B follow directly from the description of $\varphi(\ell)[e_j]$, for leaf ℓ, which was given in Definition 4.12. The conclusion follows from the definition of φ' (4.3). ∎

Fact 4.14. *For $j \in [t]$, there is only one most parsimonious labeling φ'_{e_j} which extends leaf labeling φ_{e_j} of $\mathcal{B}$.*

Proof. Recall that most parsimonious labelings minimize the sum of Hamming distances between adjacent vertices in the tree. The most parsimonious labeling obtained from Fitch's algorithm has

$$\sum_{uv \in E(\mathcal{B})} H(\varphi'_{e_j}(u), \varphi'_{e_j}(v)) = 1.$$

Because there is only one leaf ℓ with $\varphi_{e_j}(\ell) = 1$, the φ'_{e_j} obtained from Fitch's algorithm is the only extension of φ_{e_j} with the sum of Hamming distances equal 1. ∎

Fix $j \in [n]$. Next we consider the most parsimonious labelings for φ_{x_j} on $\mathcal{B}$. The same arguments will hold for each φ_{y_j}.

Run Part 1 of Fitch's algorithm on $\mathcal{B}$ with leaf labeling φ_{x_j}. For those clauses c_i which contain variable v_j, we have the following result.

Proposition 4.15 (Miklós, Kiss, and Tannier (2014))**.** *Fix a clause c_i. Suppose variable v_j is in c_i with coordinate x_j corresponding to variable v_j. Let r^i be the root of unit subtree $\mathcal{U}_i$ for c_i. Run Fitch's algorithm on $\mathcal{U}_i$ with leaf labeling φ_{x_j}. The following hold:*

1. *$B(r^i) = \{0, 1\}$.*
2. *For $u, v \in V(U_i)$, if v is a child of u, then $B(v) = \{0, 1\} \Rightarrow B(u) = \{0, 1\}$.*

In a single copy of $S(a, b, c)$, all vertices of the same distance from the root either have $B(v) \in \{\{0\}, \{1\}\}$ or all of them have $B(v) = \{0, 1\}$. This fact together with Proposition 4.15 implies that, when Fitch's algorithm is run on $\mathcal{B}$ with leaf-labeling $\varphi(x_i)$, for any $u, v \in V(T)$ with v a child of u,

$$B(v) = \{0, 1\} \Rightarrow B(u) = \{0, 1\}.$$

With this result and the structure of each $S(a, b, c)$, By Lemma 4.3, we can conclude that Fitch's algorithm finds all most parsimonious labelings of $\mathcal{B}$ that extend φ_{x_j}. Further, $B(\rho) = \{0, 1\}$ implies there are exactly two such most parsimonious labelings.

As mentioned earlier, these results also hold for coordinate y_i. Fitch's algorithm finds the only two most parsimonious labelings that extend φ_{y_j} on $\mathcal{B}$.

For most parsimonious labeling φ' that extends φ, on each $v \in V(T)$, notate $\varphi'(v)[x_j]$ by φ'_{x_j}. Likewise, define the notations φ'_{y_j} and φ'_{e_s}.

Lemma 4.16. *For leaf labeling φ of $\mathcal{B}$, Fitch's algorithm finds all most parsimonious labelings. Each is characterized by the string it assigns to the root ρ of $\mathcal{B}$ and there are precisely 2^{2n} most parsimonious labelings, one for each root label in $\{0,1\}^{2n} \times \{0\}^t$.*

Proof. Given a most parsimonious labeling φ' that extends φ, each φ'_{x_j}, φ'_{y_j}, and φ'_{e_s} is a most parsimonious labeling for that coordinate. So it suffices to first find all most parsimonious scenarios for the leaf labelings $\varphi_{x_j}, \varphi_{y_j}, \varphi_{e_s}$ for all $j \in [n]$ and $s \in [t]$ and take combinations of these labelings.

We have already seen that Fitch's algorithm will find all most parsimonious labelings for φ_{x_j} and φ_{y_j}, and there are exactly 2 of each. Fitch's algorithm will also find the one and only most parsimonious labeling for φ_{e_s}. Therefore, there are 2^{2n} most parsimonious labelings of $\mathcal{B}$ that extend φ. Part 2 of Fitch's algorithm shows that each most parsimonious labeling is characterized by the string it assigns to ρ. Since $B(\rho) = \{0,1\}$ for each φ_{x_j} and φ_{y_j} and $B(\rho) = \{0\}$ for each φ_{e_s}, the possible strings for $\varphi'(\rho)$ are $\{0,1\}^{2n} \times \{0\}^t$. ∎

Set $\mathcal{M} := \{0,1\}^{2n} \times \{0\}^t$.

Definition 4.17. *There is a bijection between $\mathcal{M}$ and the possible truth assignments for $\Psi(\Gamma)$. In particular, given any $\mu \in \mathcal{M}$, define a truth assignment for variables $\{v_i\}_{i=1}^n \cup \{w_i\}_{i=1}^n$ as follows:*

- *For each $i \in [n]$, let v_i be assigned the value true if $\mu[x_i] = 1$ and false otherwise.*
- *For each $i \in [n]$, let w_i be assigned the value true if $\mu[y_i] = 1$ and false otherwise.*

Define $\mathcal{M}_{\Psi(\Gamma)}$ to be the set of $\mu \in \mathcal{M}$ which correspond to satisfying truth assignments for $\Psi(\Gamma)$. Likewise, for any Θ, a clause or conjunction of clauses from $\Psi(\Gamma)$, define $\mathcal{M}_\Theta$ to be the set of $\mu \in \mathcal{M}$ which correspond to satisfying truth assignments for Θ.

Now we know that each most parsimonious labeling of $\mathcal{B}$ extending φ is found using Fitch's algorithm and is characterized the binary string it assigns to the root.

From here, we are interested in the number of SCJ scenarios admitted by each of these most parsimonious scenarios. Ultimately, we wish to make a distinction between the binary strings in $\mathcal{M}_{\Psi(\Gamma)}$ and those in $\mathcal{M} \setminus \mathcal{M}_{\Psi(\Gamma)}$ by examining the number of SCJ scenarios admitted by the corresponding most parsimonious labeling.

Let φ' be a most parsimonious labeling for $\mathcal{B}$. The number of scenarios which are admitted by φ' is precisely

$$\mathcal{H}(\varphi'(\rho)) := \prod_{uv \in E(\mathcal{B})} H(\varphi'(u), \varphi'(v))!.$$

To calculate this, we partition the edges of $\mathcal{B}$ into 4 sets.

First, consider the edges of the comb which connects $\{\mathcal{B}_i\}_{i=1}^{k+4n}$ to form $\mathcal{B}$. Part 2 of Fitch's algorithm will set $\varphi'(\rho) = \varphi'(\rho^i)$ where ρ^i is the root of $\mathcal{B}_i$. So the Hamming distance along each of these edges is 0.

Next we look within each $\mathcal{B}_i$.

Claim 4.18. *Set $\Phi := \bigwedge \Phi_\iota$. For $i \in [k+4n]$, let ρ^i be the root of $\mathcal{B}_i$ with children ρ^i_ℓ and ρ^i_r. Set $\eta := \varphi'(\rho^i)$. If $\eta \in \mathcal{M}'_\Phi$, then*

$$H(\eta, \varphi'(\rho^i_\ell)) = H(\eta, \varphi'(\rho^i_r)) = n - 3.$$

Otherwise

$$(n-3)!^2 \leq H(\eta, \varphi'(\rho^i_\ell))! \cdot H(\eta, \varphi'(\rho^i_r))! \leq (2n-6)!0!.$$

Proof. Suppose $\eta \in \mathcal{M}'_\Phi$. Then for each $j \in [n]$ considered in Step 2 of Definition 4.11 (there are $n-3$ such j) , if $\eta[x_j] = 0$ then we have the following properties:

- $\eta[y_j] = 1$ because η corresponds to a satisfying assignment for Φ,
- $0 = \eta[x_j] \neq \varphi'(\rho^i_r)[x_j] = 1$,
- $1 = \eta[y_j] \neq \varphi'(\rho^i_\ell)[y_j] = 0$.

On the other hand, if $\eta[x_j] = 1$ then we have the following properties:

- $\eta[y_j] = 0$ because η corresponds to a satisfying assignment for Φ,
- $1 = \eta[x_j] \neq \varphi'(\rho_\ell^i)[x_j] = 0$,
- $0 = \eta[y_j] \neq \varphi'(\rho_r^i)[y_j] = 1$.

For each $s \in [t]$, $\eta[e_s] = \varphi'(\rho_\ell^i)[e_s] = \varphi'(\rho_r^i)[e_s] = 0$. For each $j \in [n]$ which was not considered in Step 2 of Definition 4.11, $\eta[x_j] = \varphi'(\rho_\ell^i)[x_j] = \varphi'(\rho_r^i)[x_j]$ and $\eta[y_j] = \varphi'(\rho_\ell^i)[y_j] = \varphi'(\rho_r^i)[y_j]$ because the B values (from Fitch's algorithm) for these coordinates at these vertices will be $\{0, 1\}$. Thus

$$H(\eta, \varphi'(\rho_\ell^i)) = H(\eta, \varphi'(\rho_r^i)) = n - 3.$$

Alternatively, if $\eta \notin \mathcal{M}_\Phi$, then $H(\eta, \varphi'(\rho_\ell^i)) + H(\eta, \varphi'(\rho_r^i)) = 2n - 6$ because each x_i and each y_i will contribute 1 to one of the Hamming distances. Using the convexity of the factorial, this establishes the last line of the claim. ∎

Based on the construction of $S(a, b, c)$, the Hamming distance $H(\varphi'(u), \varphi'(v))$ for each edge uv in each copy of $S(a, b, c)$ is exactly 1.

The only piece remaining is $\mathcal{T}_i$. We make the following remarks for the clause $c_i = v_1 \vee v_2 \vee v_3$ to make the explanation easier. However, the arguments can be extended for any clause c_i in $\Psi(\Gamma)$.

Fact 4.19. *If t^i is the root of $\mathcal{T}_i$ and r^i is the root of one of the copies of $\mathcal{U}_i$ below $\mathcal{T}_i$, then running Fitch's algorithm for each coordinate, we find*

- $B(t^i) = B(r^i) = \{0, 1\}$ *for each x_i, $i \in [3]$, by Proposition 4.15.*
- $B(t^i) = B(r^i) = \{0\}$ *for each x_i, $i \geq 4$, by the construction of $\mathcal{B}_i$.*
- $B(t^i) = B(r^i) = \{0\}$ *for each y_i, $i \in [n]$, by the construction of $\mathcal{B}_i$.*
- $B(t^i) = B(r^i) = \{0\}$ *for each e_s, $s \in [t]$, because there is only one leaf $\ell \in L(\mathcal{B})$ with $\varphi(\ell)[e_s] = 1$.*

Therefore, it is easy to see that, along the edges of the comb which connect the copies of $\mathcal{U}_i$, the Hamming distances will be 0.

Next we turn our attention to a single copy of $\mathcal{U}_i$, say the j^{th} copy.

Fact 4.20. *Fix Γ and build binary tree $\mathcal{B}$. Fix a most parsimonious labeling φ' which extends leaf labeling φ. For clause $c_i = v_1 \vee v_2 \vee v_3$, we have the following characteristics for each $v \in \mathcal{U}_i$,*

- *for $s \geq 4$, $\varphi'(v)[x_s] = 0$,*
- *for $s \in [n]$, $\varphi'(v)[y_s] = 0$,*
- *for $s \notin E_{ij}$, $\varphi'(v)[e_s] = 0$.*

Therefore, only the values of $\varphi'(v)$ on the coordinates x_1, x_2, x_3 and e_s for $s \in E_{ij}$ will affect the Hamming distances along the edges in $\mathcal{U}_i$. These are precisely the 151 coordinates that appeared in the originally labeling $\hat{\varphi}$ of the leaves of $\mathcal{U}_i$ given by Miklós, Kiss, and Tannier (2014). For each $v \in V(\mathcal{U}_i)$, define $\hat{\varphi}'(v) : V(\mathcal{U}_i) \to \{0,1\}^{151}$ to be the restriction of $\varphi'(v)$ to these 151 coordinates. In particular, $\hat{\varphi}'$ is a most parsimonious labeling on $\mathcal{U}_i$ which extends leaf labeling $\hat{\varphi}$.

The following fact is a consequence of Fact 4.20.

Fact 4.21. *Let r^i be the root of $\mathcal{U}_i$. If $\varphi'(r^i) = \hat{\varphi}'(r^i)$, then for each $uv \in E(\mathcal{U}_i)$,*

$$H(\varphi'(u), \varphi'(v)) = H(\hat{\varphi}'(u), \hat{\varphi}'(v)).$$

As a result

$$\prod_{uv \in \mathcal{U}_i} H(\varphi'(u), \varphi'(v))! = \prod_{uv \in \mathcal{U}_i} H(\hat{\varphi}'(u), \hat{\varphi}'(v))!.$$

This is calculated as follows:

Fact 4.22 (Miklós, Kiss, and Tannier (2014))**.** *Fix $i \in [k + 4n]$, the binary tree $\mathcal{U}_i$ with root r^i, and leaf-labeling $\hat{\varphi}$. Then for any most parsimonious labeling $\hat{\varphi}'$ which extends $\hat{\varphi}$:*

1. *If $\hat{\varphi}'(r^i)$ corresponds to a satisfying truth assignment for c_i, then*

$$\prod_{uv \in \mathcal{U}_i} H(\hat{\varphi}'(u), \hat{\varphi}'(v))! = 2^{156} \times 3^{64}.$$

2. *If $\hat{\varphi}'(r^i)$ corresponds to a truth assignment which does not satisfy c_i, then*

$$\prod_{uv \in \mathcal{U}_i} H(\hat{\varphi}'(u), \hat{\varphi}'(v))! = 2^{136} \times 3^{76}.$$

Since $\varphi'(\rho) = \varphi'(r^i) = \hat{\varphi}'(r^i)$, $\varphi'(\rho)$ corresponds to a satisfying truth assignment for c_i if and only if $\hat{\varphi}'(r^i)$ also corresponds to a satisfying truth assignment for c_i.

As a result of the above discussion, we have proven the following claim.

Claim 4.23. *Fix $i \in [k + 4n]$. If $\varphi'(\rho)$ corresponds to a satisfying truth assignment for clause c_i and $\bigwedge_{\iota \in [n]} \Phi_\iota$, then*

$$\prod_{uv \in E(\mathcal{B}_i)} H(\varphi'(u), \varphi'(v))! = (n-3)!^2 \left(2^{156} \times 3^{64}\right)^{16n^2+8kn}.$$

If $\varphi'(\rho)$ corresponds to a truth assignment which does not satisfy c_i, then

$$\begin{aligned}(n-3)!^2 \left(2^{136} \times 3^{76}\right)^{16n^2+8kn} &\leq \prod_{uv \in E(\mathcal{B}_i)} H(\varphi'(u), \varphi'(v))! \\ &\leq (2n-6)! \left(2^{136} \times 3^{76}\right)^{16n^2+8kn}.\end{aligned}$$

If $\varphi'(\rho)$ corresponds to a truth assignment which satisfies c_i but does not satisfy $\bigwedge_{i \in [n]} \Phi_i$, then

$$\begin{aligned}(n-3)!^2 \left(2^{156} \times 3^{64}\right)^{16n^2+8kn} &< \prod_{uv \in E(\mathcal{B}_i)} H(\varphi'(u), \varphi'(v))! \\ &\leq (2n-6)! \left(2^{156} \times 3^{64}\right)^{16n^2+8kn}.\end{aligned}$$

Observe,

$$\begin{aligned}
\frac{(2n-6)!\left[2^{136}\times 3^{76}\right]^{16n^2+8kn}}{(n-3)!^2\left[2^{156}\times 3^{64}\right]^{16n^2+8kn}} &= \left[\frac{3^{12}}{2^{20}}\right]^{16n^2+8kn}\binom{2n-6}{n-3}\\
&< \left[\frac{3^{12}}{2^{20}}\right]^{16n^2+8kn} 2^{2n}\\
&< \left[\frac{3^{12}}{2^{20}}\right]^{16n^2+8kn} 2^{2n+k}\\
&= \left[\frac{3^{12}}{2^{20-1/(8n)}}\right]^{16n^2+8kn}\\
&< \left[\frac{3^{12}}{2^{19.5}}\right]^{16n^2+8kn}\\
&< 1.
\end{aligned}$$

Consequently,

$$\begin{aligned}
(2n-6)!\left(2^{136}\times 3^{76}\right)^{16n^2+8kn} &< (n-3)!^2\left(2^{156}\times 3^{64}\right)^{16n^2+8kn}\\
&< (2n-6)!\left(2^{156}\times 3^{64}\right)^{16n^2+8kn}.
\end{aligned}$$

Claim 4.24. *If $\varphi'(\rho)$ corresponds to a satisfying truth assignment for $\Psi(\Gamma)$, then*

$$\mathcal{H}(\varphi'(\rho)) = \left[(n-3)!^2\left(2^{136}\times 3^{76}\right)^{16n^2+8kn}\right]^{k+4n} =: B_{good}.$$

If $\varphi'(\rho)$ corresponds to a truth assignment which does not satisfy $\Psi(\Gamma)$, then there must be a clause c_i for some $i \in [k+4n]$ which is not satisfied. Therefore,

$$\begin{aligned}
\mathcal{H}(\varphi'(\rho)) \leq & (2n-6)!\left(2^{136}\times 3^{76}\right)^{16n^2+8kn}\\
& \cdot\left[(2n-6)!\left(2^{156}\times 3^{64}\right)^{16n^2+8kn}\right]^{k+4n-1}\\
=: & B_{bad}.
\end{aligned}$$

Define

$$B_{total} := \sum_{\varphi'} \mathcal{H}(\varphi'(\rho))$$

which is the total number of most parsimonious SCJ scenarios for $\mathcal{B}$ which extend leaf labeling φ, as in Definition 1.22.

Given only B_{total}, we would like to determine the number of satisfying truth assignments, $|S|$, for $\Psi(\Gamma)$.

$$\begin{aligned} B_{total} &= \sum_{\eta \in \mathcal{M}'_{\Psi(\Gamma)}} \mathcal{H}(\eta) + \sum_{\eta' \in \mathcal{M} \setminus \mathcal{M}'_{\Psi(\Gamma)}} \mathcal{H}(\eta') \\ &= |S| B_{good} + \sum_{\eta' \in \mathcal{M} \setminus \mathcal{M}'_{\Psi(\Gamma)}} \mathcal{H}(\eta'). \end{aligned}$$

As long as $\sum_{\eta' \in \mathcal{M} \setminus \mathcal{M}'_{\Psi(\Gamma)}} \mathcal{H}(\eta') < B_{good}$, we can conclude that the number of satisfying truth assignments for $\Psi(\Gamma)$ (and for Γ) is precisely

$$\left\lfloor \frac{B_{total}}{B_{good}} \right\rfloor.$$

Observe, for $n \geq 2$,

$$\begin{aligned} \frac{2^{2n} B_{bad}}{B_{good}} &= 2^{2n} \left[\frac{3^{12}}{2^{20}} \right]^{16n^2+8kn} \binom{2n-6}{n-3}^{k+4n} \\ &< 2^{2n} \left[\frac{3^{12}}{2^{20}} \right]^{16n^2+8kn} 2^{2n(k+4n)} \\ &< 2^{8n^2+2kn+2n} \left[\frac{3^{12}}{2^{20}} \right]^{16n^2+8kn} \\ &< 2^{8n^2+4kn} \left[\frac{3^{12}}{2^{20}} \right]^{16n^2+8kn} \\ &= \left[\frac{3^{12}}{2^{20-1/2}} \right]^{16n^2+8kn} \\ &< 1. \end{aligned}$$

Because there are only 2^{2n} truth assignments and 2^{2n} most parsimonious labelings, we obtain our desired result:

$$\sum_{\eta' \in U} \mathcal{H}(\eta') \leq \sum_{\eta' \in U} B_{bad} \leq 2^{2n} B_{bad} < B_{good}.$$

Therefore, if we could determine the total number of most parsimonious scenarios for this binary tree in polynomial time, then we could obtain the total number of satisfying assignments for $\Psi(\Gamma)$ and for Γ in polynomial time. This completes the proof. ∎

Chapter 5

Eccentricity Sums in Trees

The *eccentricity* of a vertex v in a connected graph G is defined in terms of the distance function as

$$\mathrm{ecc}_G(v) := \max_{u \in V(G)} d(u, v).$$

The radius of G, $\mathrm{rad}(G)$, is the minimum eccentricity while the diameter, $\mathrm{diam}(G)$, is the maximum. The center, $C(G)$, is the collection of vertices whose eccentricity is exactly $\mathrm{rad}(G)$.

We focus our attention on trees, where the center has at most two vertices (Jordan 1869) and the diameter is realized by a leaf. We also explore the *total eccentricity* of a tree T, defined as the sum of the vertex eccentricities:

$$\mathrm{Ecc}(T) := \sum_{z \in V(T)} \mathrm{ecc}_T(z).$$

For a fixed tree T with $v \in C(T)$ and any $z \in V(T)$,

$$\min_{u \in L(T)} \frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)} \leq \frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(z)} \leq \frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)}$$

where $L(T)$ denotes the leaf set of T. This motivates the study in Section 2 of the extremal values and structures for the following ratios where $u, w \in L(T)$ and $v \in C(T)$,

$$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)}, \quad \frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)}, \quad \frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(v)}, \quad \text{and} \quad \frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(w)}.$$

The results are analogous to similar studies on distance in (Barefoot, Entringer, and Székely 1997) and on the number of subtrees in (Székely and Wang 2014; Székely and

Wang 2013). As in those papers, the behavior of ratios is more delicate than that of their numerators or denominators.

For a graph with n vertices, the total eccentricity is n times the average eccentricity. Dankelmann and Mukwembi (2014) gave sharp upper bounds on the average eccentricity of graphs in terms of independence number, chromatic number, domination number, as well as connected domination number. For trees with n vertices, Dankelmann, Goddard, and Swart (2004) showed that the path maximizes $\mathrm{Ecc}(T)$. In Section 3, we prove that the star minimizes $\mathrm{Ecc}(T)$ among trees with a given order. Turning our attention to trees with a fixed degree sequence, we prove that the "greedy" caterpillar maximizes $\mathrm{Ecc}(T)$ while the "greedy" tree minimizes $\mathrm{Ecc}(T)$. This provides further information about the total eccentricity of "greedy" trees across degree sequences.

From here forward, we assume that T is a tree with n vertices. Given two vertices $a, b \in V(T)$, $P(a, b)$ will be the unique path between a and b in T.

5.1 Extremal ratios

In this section, we consistently use the letters u, w to denote leaf vertices while v is a center vertex. Before delving into ratios, the following observation from Jordan (1869) is given without proof, and will be used many times. The next observation is a simple calculation which will be useful in our proofs.

Observation 5.1. *The center, $C(T)$, contains at most 2 vertices. These vertices are located in the middle of a maximum length path, P. If $\{v\} = C(T)$, v divides P into two paths, each of length* $\mathrm{rad}(T)$. *If $\{v, z\} = C(T)$, the removal of $vz \in E(T)$ will divide P into two paths, each of length* $\mathrm{rad}(T) - 1$.

Observation 5.2. *For any path P with y edges and $y+1$ vertices,*

$$\mathrm{Ecc}(P) = \sum_{z \in V(P)} \mathrm{ecc}_P(z) = \begin{cases} \frac{3}{4}y^2 + y & \text{if } y \text{ is even} \\ \frac{3}{4}y^2 + y + \frac{1}{4} & \text{if } y \text{ is odd.} \end{cases}$$

5.1.1 On the extremal values of $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)}$ where $v \in C(T)$

Theorem 5.3. *Let T be a tree with $n \geq 2$ vertices. For any $v \in C(T)$, we have*

$$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)} \leq 2n - 1.$$

For $n \geq 3$, equality holds if and only if T is a star centered at v.

Proof. Let T be an arbitrary tree with $v \in C(T)$. It is known that for any tree T, $\mathrm{diam}(T) \leq 2\,\mathrm{rad}(T)$ and for any vertex $z \in V(T)$, $\mathrm{rad}(T) \leq \mathrm{ecc}_T(z) \leq \mathrm{diam}(T)$. Because $\mathrm{ecc}_T(v) = \mathrm{rad}(T)$, the bound in the theorem is proved as follows:

$$\mathrm{Ecc}(T) \leq \mathrm{ecc}_T(v) + (n-1)\,\mathrm{diam}(T) \leq (2n-1)\,\mathrm{rad}(T).$$

Equality holds precisely when T has $\mathrm{ecc}_T(z) = 2\,\mathrm{ecc}_T(v)$ for all vertices $z \neq v$. Because the eccentricities of adjacent vertices differ by at most 1, $\mathrm{ecc}_T(v) = 1$ and $\mathrm{ecc}_T(z) = 2$ for all $z \neq x$ which is only true for the star. ∎

Theorem 5.4. *Let T be a tree with $n \geq 2$ vertices. Let k and i be nonnegative integers with $0 \leq i \leq 2k$ and $n = k^2 + i$. For any $v \in C(T)$, we have*

$$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)} \geq \begin{cases} n - 3 + 2k + \frac{i}{k} & \text{if } 0 \leq i \leq k \\ n - 3 + 2k + \frac{i+1}{k+1} & \text{if } k+1 \leq i \leq 2k. \end{cases}$$

For $n \geq 4$, equality holds if and only if T is a tree whose longest path has $2x$ vertices ($x = k$ in the first case and $x = k+1$ in the second) and each other vertex is adjacent to one of the two center vertices of this path. For $i = k$, the two bounds agree and both values for x provide an extremal tree.

Proof. Let T be a tree with $n \geq 3$ vertices and let $v \in C(T)$. If T is a star then $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)} = 2n - 1$ which is strictly greater than the bounds in the theorem.

For the remainder of the proof, we consider the case when T is not a star. By Observation 5.1, there is a maximum-length path $P := P(u, w)$ with v in the middle and $d(u, v) = \mathrm{ecc}_T(v)$. We now consider two cases, based upon the size of $C(T)$.

If $C(T) = \{v\}$, then both $P(u, v)$ and $P(v, w)$ have length $\mathrm{ecc}_T(v)$. Let S be the non-empty set $\{w' \in L(T) : w' \neq u \text{ and } d(v, w') = \mathrm{ecc}_T(v)\}$. Create a new tree F from T by detaching each leaf $w' \in S$ and appending each one to v. This tree is different from T because T was not a star. For any $z \in V(T)$, $\mathrm{ecc}_T(z) \geq \mathrm{ecc}_F(z)$. Further, for each $w' \in S$, $\mathrm{ecc}_T(w') > \mathrm{ecc}_F(w')$. As a result, $\mathrm{Ecc}(T) > \mathrm{Ecc}(F)$. As for $v \in C(T)$, $\mathrm{ecc}_F(v) = \mathrm{ecc}_T(v) = d_T(u, v)$ because $u \notin S$. The length of the longest path in F is one less than the length of the longest path in T which implies $v \in C(F)$ and $|C(F)| = 2$. Altogether, we see $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)} > \frac{\mathrm{Ecc}(F)}{\mathrm{ecc}_F(v)}$. Hence, to minimize $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)}$, it suffices to consider those trees with two center vertices.

Suppose $|C(T)| = 2$ and let $x := \mathrm{ecc}_T(v)$. Here, the path P has length $2x - 1$ and the vertices on P realize their eccentricities along this path since it has maximum length. Explicitly calculating the eccentricities of the vertices on P, using Observation 5.2, and lower bounding all other eccentricities by $x + 1$, we have

$$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)} \geq \frac{1}{x}\left(\left(3x^2 - x\right) + (n - 2x)(x + 1)\right) = x + (n - 3) + \frac{n}{x} =: f(x)$$

Equality holds if and only if each vertex not on P is a neighbor of one of the center vertices of P, as in Fig. 5.1.

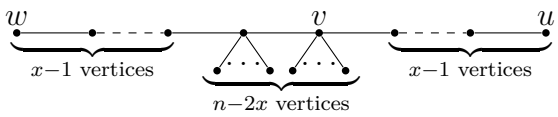


Figure 5.1 A tree which minimizes $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)}$.

To determine the value of x which minimizes $f(x)$, we use the first derivative test.

Because $f'(x) = 1 - \frac{n}{x^2}$ is negative for $x < \sqrt{n}$ and positive for $x > \sqrt{n}$, the minimum of $f(x)$ is obtained when

$$x \in \{\lfloor \sqrt{n} \rfloor, \lceil \sqrt{n} \rceil\} \subseteq \{k, k+1\}.$$

Because $f(k+1) - f(k) = \frac{k-i}{k(k+1)}$, $f(k) \leq f(k+1)$ precisely when $i \geq k$ with equality when $i = k$, as stated in the theorem. ∎

5.1.2 On the extremal values of $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)}$ where $u \in L(T)$

Theorem 5.5. *Let T be a tree on $n \geq 8$ vertices. Let k and i be integers with $0 \leq i \leq 2k$ and $2n - 1 = k^2 + i$. For any $u \in L(T)$, we have*

$$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)} \leq \begin{cases} 2n + 1 - 2k - \frac{i}{k} & \text{if } 0 \leq i \leq k \\ 2n + 1 - 2k - \frac{i+1}{k+1} & \text{if } k+1 \leq i \leq 2k. \end{cases}$$

Equality holds if and only if T is a tree with longest path $P = z_1 z_2 \dots z_{2x-1}$ ($x = k$ in the first case and $x = k+1$ in the second), leaf u adjacent to z_x, and each other vertex adjacent to either z_2 or z_{2x-2}. For $i = k$, the two bounds agree and both values of x will provide an extremal tree.

Proof. Let T be a tree with $n \geq 8$ vertices and $u \in L(T)$. If T is a path, then $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)} \leq \frac{3}{4}n + \frac{1}{2}$ which is strictly smaller than the bounds in the theorem.

For the remainder of the proof, we will suppose T is not a path. Fix $P := P(w, w')$ to be a maximum-length path in T. For any leaf $u \in L(T)$ different from w and w', $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(w)} \leq \frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)}$. Because we are interested in an upper bound, it suffices to consider leaves u which are not on P.

Let u be a leaf of T which is not on P and let $x := \mathrm{ecc}_T(u)$. There is a unique path from u to the closest vertex on P, say z. Then $d(u, w) = d(u, z) + d(z, w)$ and $d(u, w') = d(u, z) + d(z, w')$. Since $d(u, w)$ and $d(u, w')$ are at most x and $d(u, z) \geq 1$, we have $d(w, w') = d(w, z) + d(z, w') \leq 2x - 2$. Because P has maximum length, every

vertex on P realizes its eccentricity along P. Every vertex not on P has eccentricity at most $2x-2$. This gives the upper bound

$$\begin{aligned}\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)} &\leq \frac{1}{x}\left(x + \left(\frac{3}{4}(2x-2)^2 + (2x-2)\right) + (n-2x)(2x-2)\right) \\ &= \frac{-x^2 + (2n+1)x - (2n-1)}{x}.\end{aligned}$$

Equality is achieved precisely when T is a tree with longest path P on $2x-1$ vertices, u is adjacent to the middle vertex of P, and all other vertices have eccentricity $2x-2$. Such a tree T is shown in Fig. 5.2.

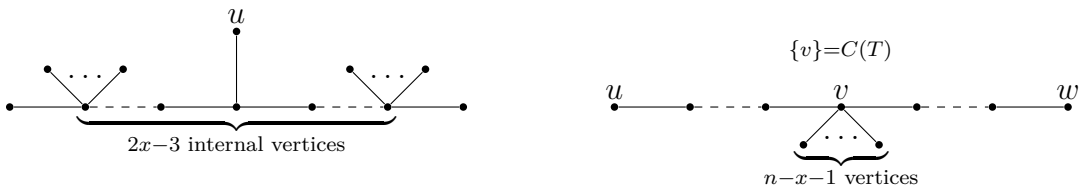


Figure 5.2 A tree (left) which maximizes $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)}$ and a tree (right) which minimizes $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)}$.

It remains to determine the value of x that will maximize $\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)}$ for trees with the structure described above. The first derivative test shows that $f(x)$ is maximized when

$$x \in \{\left\lfloor \sqrt{2n-1} \right\rfloor, \left\lceil \sqrt{2n-1} \right\rceil\} \subseteq \{k, k+1\}.$$

The larger of $f(k)$ and $f(k+1)$ gives the appropriate upper bound in (5.5). In addition, we must require $2x \leq n$ in order to have a realizable tree. One can individually check that this is the case for $n \in \{8, 9, \ldots, 12\}$. When $n \geq 13$, we have $k \geq 5$ in which case $0 \leq k^2 - 4k - 3$ which implies $2x \leq 2(k+1) \leq n$. ∎

Theorem 5.6. *Let T be a tree of order $n \geq 5$. Let k and i be nonnegative integers with $0 \leq i \leq 2k$ and $4n - 4 = k^2 + i$. Then for any leaf u,*

$$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)} \geq \begin{cases} \frac{n-1}{2} + \frac{k}{2} + \frac{i}{4k} & \text{if } k \text{ is even} \\ \frac{n-1}{2} + \frac{k}{2} + \frac{i+1}{4(k+1)} & \text{if } k \text{ is odd.} \end{cases} \tag{5.1}$$

Equality holds if and only if T is a tree with longest path P of length $2x$ ($2x = k$ for the first case and $2x = k + 1$ for the second) with all other vertices adjacent to the middle vertex of P as shown in Fig. 5.2. When $i = k$, both bounds in (5.1) *give the same value and both give extremal structures.*

Proof. Let T be a tree and $u \in L(T)$. Let $x := \mathrm{ecc}_T(u)$ and choose $w \in L(T)$ so that $d(u, w) = x$. Let $P := P(u, w)$. The vertices on P have $\mathrm{ecc}_T(u) \geq \mathrm{ecc}_P(u)$. The eccentricity of any vertex not on P is at least $1 + \frac{x}{2}$ with equality if x is even and these vertices are adjacent to the center vertex of P. This gives the following lower bound:

$$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)} \geq \frac{1}{x}\left(\left(\frac{3}{4}x^2 + x\right) + (n - x - 1)\left(1 + \frac{x}{2}\right)\right) =: f(x)$$

where equality holds when P has even length and all vertices not on P are adjacent to the center vertex of P as in Fig. 5.2. Examination of $f'(x)$ shows that the ratio is minimized when

$$x \in \left\{\left\lfloor\sqrt{4n-4}\right\rfloor, \left\lceil\sqrt{4n-4}\right\rceil\right\} \subseteq \{k, k+1\}.$$

We already established that the lower bound is tight for even x. For the universal lower bound, we let $x = k$ if k is even and $k+1$ otherwise. Both will yield a realizable tree because $x \leq n - 1$ for $n \geq 5$. It is also important to note that if $4n - 4$ is a perfect square, then $k = \lfloor\sqrt{4n-4}\rfloor = \lceil\sqrt{4n-4}\rceil = 2\sqrt{n-1}$, an even value. The lower bounds in (5.1) are exactly $f(k)$ and $f(k + 1)$. For thoroughness, it can be verified that $f(k) \leq f(k + 2)$ and $f(k + 1) \leq f(k - 1)$, for $k > 1$ to show that our choice of the even integer nearest $\sqrt{4n - 4}$ was correct for this concave up function.

For thoroughness, it is shown below that $f(k) \leq f(k+2)$ and $f(k+1) \leq f(k-1)$, for $k > 1$. (One can quickly check that the bound also holds for $n = 3$ when $k = 1$.) Thus our choice of the even integer nearest $\sqrt{4n - 4}$ was correct for this concave up function.

$$\begin{aligned} f(k+2)-f(k) &= \frac{n-1}{k+2}+\frac{1}{2}-\frac{n-1}{k} \\ &= \frac{2k(n-1)+k(k+2)-2(k+2)(n-1)}{2k(k+2)} \\ &= \frac{k^2+2k-(4n-4)}{2k(k+2)} \\ &= \frac{k^2+2k-(k^2+i)}{2k(k+2)} \\ &\geq 0. \end{aligned}$$

$$\begin{aligned} f(k-1)-f(k+1) &= -\frac{1}{2}+\frac{n-1}{k-1}-\frac{n-1}{k+1} \\ &= \frac{-(k-1)(k+1)+2(n-1)(k+1)-2(n-1)(k-1)}{2(k-1)(k+1)} \\ &= \frac{-k^2+1+4(n-1)}{2(k-1)(k+1)} \\ &= \frac{-k^2+1+k^2+i}{2(k-1)(k+1)} \\ &= \frac{i+1}{2(k-1)(k+1)} \\ &> 0. \end{aligned}$$

■

5.1.3 On the extremal values of $\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(v)}$ where $u \in L(T)$, $v \in C(T)$

Theorem 5.7. *Let T be a tree on $n \geq 3$ vertices with $u \in L(T)$ and $v \in C(T)$. Then*

$$\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(v)} \leq 2,$$

where the upper bound is tight for stars, even length paths, and more. If, in addition, $n \geq 5$, then

$$1+\frac{1}{\left\lfloor \frac{n-1}{2} \right\rfloor} \leq \frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(v)}.$$

Equality holds if and only if T is one of the following trees: (1) For any $n \geq 5$, T has a longest path P on $n-1$ vertices with a single vertex u adjacent to $v \in C(P)$. (2) For even $n \geq 5$, T has a longest path P on $n-2$ vertices with u adjacent to $v \in C(P)$ and w adjacent to any internal vertex of P. These structures are drawn in Fig. 5.3.

Proof. The upper bound of 2 follows from the facts that $\operatorname{rad}(T) = \operatorname{ecc}_T(v)$ and $\operatorname{ecc}_T(u) \leq \operatorname{diam}(T) \leq 2\operatorname{rad}(T)$. This bound is tight for all trees whose maximum-length path has an odd number of vertices and u a leave of one of these paths.

Turning our attention to the lower bound, let T be a tree with $n \geq 5$ vertices. We first show that it holds for paths. If T is a path, then

$$\frac{\operatorname{ecc}_T(u)}{\operatorname{ecc}_T(v)} \geq \frac{n-1}{\frac{n}{2}} = 1 + \frac{n-2}{n} \geq 1 + \frac{1}{\left\lfloor \frac{n-1}{2} \right\rfloor}.$$

For the remainder of the proof, we assume T is not a path. Because $n \geq 5$, $\operatorname{ecc}_T(u) \geq \operatorname{ecc}_T(v) + 1$ with equality exactly when $uv \in E(T)$. In addition, because $v \in C(T)$, Observation 5.1 guarantees a maximum-length path P with v in the middle. Because T is not a path, P has at most $n-1$ vertices and $\operatorname{ecc}_T(v) \leq \lceil \frac{n-2}{2} \rceil$. These two inequalities result in the desired bound.

$$\frac{\operatorname{ecc}_T(u)}{\operatorname{ecc}_T(v)} \geq 1 + \frac{1}{\operatorname{ecc}_T(v)} \geq 1 + \frac{1}{\lceil \frac{n-2}{2} \rceil} = 1 + \frac{1}{\lfloor \frac{n-1}{2} \rfloor}.$$

Finally, let us analyze the trees T for which equality holds. Because P has at most $n-1$ vertices, we first examine the necessary and sufficient conditions to have $\operatorname{ecc}_T(v) = \lceil \frac{n-2}{2} \rceil$, based on the parity of n. If n is odd, then $\operatorname{ecc}_T(v) = \frac{n-1}{2}$ if and only if P has $n-1$ vertices. For even n, $\operatorname{ecc}_T(v) = \frac{n-2}{2}$ if and only if P has $n-1$ or $n-2$ vertices.

Therefore, the bound in the theorem is tight exactly when T is one of the following two trees which are drawn in Fig 5.3: (1) T is a tree with longest path P on $n-1$ vertices and leaf u adjacent to $v \in C(P)$. (2) For even n, T is a tree with maximum-length path $P = z_1 z_2 \dots z_{n-2}$ with u adjacent to $v \in C(P)$ and w adjacent to z_i for some $i \in \{2, 3, \dots, n-3\}$. ∎

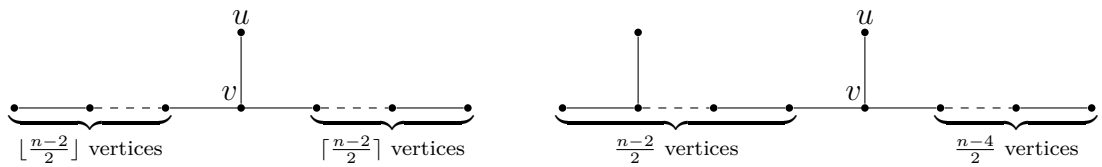


Figure 5.3 Trees which minimize $\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(v)}$, the right one for even n only.

5.1.4 On the extremal values of $\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(w)}$ where $u, w \in L(T)$

First note that since the maximum and minimum values of $\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(w)}$ are reciprocals of each other, we only consider the maximum.

Theorem 5.8. *Let T be a tree with $n \geq 4$ vertices. For any $u, w \in L(T)$, we have*

$$\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(w)} \leq 2 - \frac{2}{\lfloor \frac{n}{2} \rfloor}.$$

For even n, equality holds if and only if T is a tree with longest path $P = uz_2z_3 \dots z_{n-1}$ and leaf w adjacent to $z_{n/2}$. For odd n, equality holds if and only if T is a tree with longest path $P = uz_2z_3 \dots z_{n-2}$, leaf w adjacent to $z_{(n-1)/2}$ and leaf ω adjacent to z_i for some $i \in \{2, \dots, n-3\}$. These constructions are drawn in Fig. 5.4.

Proof. Let T be a tree and let $u, w \in L(T)$. For the upper bound, it is reasonable to assume $\mathrm{ecc}_T(u) \geq \mathrm{ecc}_T(w)$. If $d(u, w) = \mathrm{ecc}_T(u)$, then $\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(w)} = 1$ which is strictly smaller than the bound in the theorem.

For the remainder of the proof, we focus on the case where $d(u, w) < \mathrm{ecc}_T(u)$. Choose $y \in L(T)$ so that $\mathrm{ecc}_T(u) = d(u, y)$ and let $P := P(u, y)$. There is a unique path from w to the nearest vertex, say z, on P. Thus

$$\mathrm{ecc}_T(w) \geq d(w, z) + \max\{d(z, u), d(z, y)\} \geq 1 + \left\lceil \frac{1}{2} \mathrm{ecc}_T(u) \right\rceil$$

where equality holds if $d(w, z) = 1$ and $|d(z, u) - d(z, y)| \leq 1$. We now consider two cases based on the parity of $\mathrm{ecc}_T(u)$.

First suppose $x := \mathrm{ecc}_T(u)$ is odd. Let S be the collection $\{y : d(u, y) = x\}$. Notice that w is not in S because $d(u, w) < x$. Now create a new tree F from T by detaching each $y \in S$ and reattaching each as a pendant vertex adjacent to the unique neighbor of u in T. As a result, $\mathrm{ecc}_F(u) = x - 1$, an even integer. By the above argument, $\mathrm{ecc}_T(w) \geq 1 + \lceil \frac{x}{2} \rceil = 1 + \frac{1}{2}(x + 1)$ while $\mathrm{ecc}_F(w) \geq 1 + \lceil \frac{1}{2} \mathrm{ecc}_F(u) \rceil = 1 + \frac{1}{2}(x - 1)$. As a result, we obtain tight upper bounds $\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(w)} \leq \frac{x}{\frac{1}{2}(x+3)}$ and $\frac{\mathrm{ecc}_F(u)}{\mathrm{ecc}_F(w)} \leq \frac{x-1}{\frac{1}{2}(x+1)}$. The second gives the larger upper bound. Since we seek a tight universal upper bound for the ratio, it suffices to consider only trees with u having even eccentricity.

Assume $\mathrm{ecc}_T(u)$ is even. If n is even, then $\mathrm{ecc}_T(u) \leq n - 2$ because w is not on P. However, we can tighten this to $\mathrm{ecc}_T(u) \leq n - 3$ when n is odd because of our assumption about the parity of $\mathrm{ecc}_T(u)$. In either case, $1 + \frac{1}{2} \mathrm{ecc}_T(u) \leq \lfloor \frac{n}{2} \rfloor$. This give the desired bound:

$$\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(w)} \leq \frac{\mathrm{ecc}_T(u)}{1 + \frac{1}{2} \mathrm{ecc}_T(u)} = 2 - \frac{2}{1 + \frac{1}{2} \mathrm{ecc}_T(u)} \leq 2 - \frac{2}{\lfloor \frac{n}{2} \rfloor}.$$

Finally, we characterize the trees T, based on the parity of n, for which equality holds. For even n, equality holds if and only if T is a tree with longest path P on $n - 1$ vertices with leaf w adjacent to the center of P. For odd n, equality holds if and only if T is a tree with longest path P on $n - 2$ vertices with leaf w adjacent to the center of P and leaf ω adjacent to any internal vertex of P. This is exemplified by Fig. 5.4, with the additional leaf that occurs only for odd n in gray. ∎

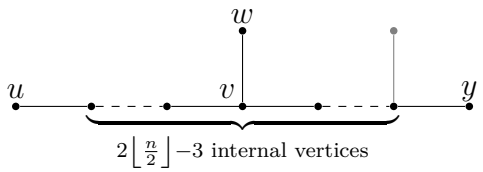
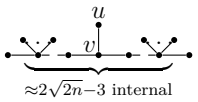
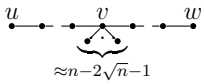
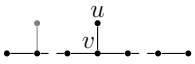
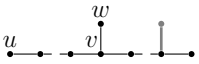


Figure 5.4 A tree which maximizes $\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(w)}$.

Table 5.1 summarizes the results in section 2. Vertex labels appear as in the theorems. Specifically v is always in $C(T)$ while each u and w are leaves of T.

Table 5.1 A summary of results for an arbitrary tree T on n vertices with $v \in C(T)$ and $u, w \in L(T)$.

	Bound	Extremal Tree
$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)}$	$\leq 2n-1$	
$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(v)}$	$\geq n+2\sqrt{n}-a(n)$	
$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)}$	$\leq 2n-2\sqrt{2n}+b(n)$	
$\frac{\mathrm{Ecc}(T)}{\mathrm{ecc}_T(u)}$	$\geq \frac{1}{2}n+\sqrt{n}-c(n)$	
$\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(v)}$	≤ 2	Stars, even length paths with pendant edges, etc.
$\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(v)}$	$\geq 1+\frac{2}{n}+O(\frac{1}{n^2})$	
$\frac{\mathrm{ecc}_T(u)}{\mathrm{ecc}_T(w)}$	$\leq 2-\frac{4}{n}+O(\frac{1}{n^2})$	

The quantities $a(n), b(n), c(n)$ are bounded as follows: $1 \leq a(n) \leq 5$, $-1 \leq b(n) \leq 5$, $0 \leq c(n) \leq \frac{3}{2}$.

5.2 Extremal structures

In this section, we fix a class of trees and find the ones in this class that maximize $\mathrm{Ecc}(T)$ and the ones that minimize $\mathrm{Ecc}(T)$. First, we consider the trees on n vertices. Then, we fix a degree sequence and search in the class of trees that realize this degree sequence.

5.2.1 General trees

For many indices, such as the sum of distances and the number of subtrees, the star and the path are extremal. Dankelmann, Goddard, and Swart (2004) showed that the path maximizes $\mathrm{Ecc}(T)$ among trees with given order. We show that the star minimizes $\mathrm{Ecc}(T)$ among trees with given order.

Proposition 5.9. *For any tree T with $n > 2$ vertices,*

$$\mathrm{Ecc}(T) \geq 1 + 2(n-1) = 2n - 1$$

with equality if and only if T is a star.

Proof. Any tree with at least three vertices has at most one vertex which is adjacent to every other vertex (hence with eccentricity 1). Thus we have

$$\mathrm{Ecc}(T) \geq 1 + 2(n-1) = 2n - 1.$$

Equality holds if and only if the single center vertex has eccentricity 1 and all other vertices have eccentricity 2. This characterizes the star. ∎

5.2.2 Trees with given degree sequences

Given a degree sequence, let $\mathcal{T}$ be the class of trees that realize this degree sequence. We determine which trees in $\mathcal{T}$ have total eccentricity equal to $\min_{T\in\mathcal{T}} \mathrm{Ecc}(T)$ or $\max_{T\in\mathcal{T}} \mathrm{Ecc}(T)$. We note that a sequence $(d_1, d_2, \ldots, d_n)$ is the degree sequence for a tree if and only if $\sum_{i=1}^{n} d_i = 2(n-1)$ and each d_i is a positive integer.

General Caterpillars

Among all trees with a given degree sequence, the sum of distances is maximized by a caterpillar (Zhang, Xiang, Xu, and Pan 2008) and the number of subtrees is minimized by a caterpillar (Sills and Wang 2015; Zhang and Zhang 2015). However, completely

characterizing the extremal caterpillar turns out to be a very difficult question in both cases. For the sum of distances, it is a quadratic assignment problem that is NP-hard in the ordinary sense and solvable in pseudo-polynomial time (Çela, Schmuck, Wimer, and Woeginger 2011).

Definition 5.10 (Wang (2014))**.** *For $n \geq 3$, let $\overline{d} = (d_1, d_2, \ldots, d_n)$ be the non-decreasing degree sequence of a tree with $d_k > 1$ and $d_{k+1} = 1$ for some $k \in [n-2]$. The greedy caterpillar, T, is constructed as follows:*

- *Start with a path $P = z_1 z_2 \ldots z_k$.*
- *Let $\phi : \{z_i\}_{i=1}^k \to \{d_i\}_{i=1}^k$ be a one-to-one function such that, for each pair $i, j \in [k]$, if $\mathrm{ecc}_P(z_i) > \mathrm{ecc}_P(z_j)$ then $\phi(z_i) \geq \phi(z_j)$.*
- *For each $i \in \{2, 3, \ldots, k-1\}$, attach $\phi(z_i) - 2$ pendant vertices to z_i. For $i \in \{1, k\}$, attach $\phi(z_i) - 1$ pendant vertices to z_i.*

Fig. 5.5 gives two examples of greedy caterpillars and highlights the fact that greedy caterpillars are not unique.

Figure 5.5 Non-isomorphic greedy caterpillars for degree sequence $(7, 6, 5, 4, 4, 1, \ldots, 1)$.

Proposition 5.11. *Among trees with a given tree degree sequence, the greedy caterpillar has the maximum total eccentricity.*

Proof. Fix a degree sequence $\overline{d} = (d_1, \ldots, d_n)$ which is written in the form described in Definition 5.10. Let $\mathcal{T}$ be the collection of trees with degree sequence $\overline{d}$. Let $T \in \mathcal{T}$ be a tree such that $\mathrm{Ecc}(T) = \max_{F \in \mathcal{T}} \mathrm{Ecc}(F)$. We first show that T is a caterpillar.

For contradiction, suppose T is not a caterpillar. Let $P_T(u,v) = uu_1u_2\dots u_kv$ be a longest path in T. Let $x \in [k]$ be the least integer such that u_x has a non-leaf neighbor w not on $P_T(u,v)$. Because P is a maximum-length path, $x \neq 1$. Let W be the component containing w in $T - \{u_xw\}$.

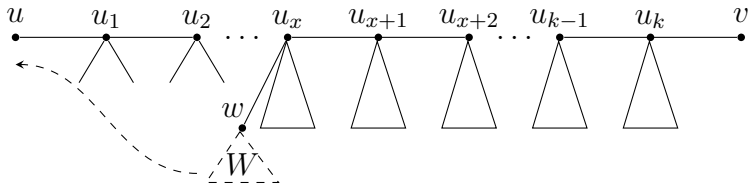


Figure 5.6 Generating T' from T for the proof of Proposition 5.11.

Create a new tree T' from T by replacing each edge of the form zw in W with the edge zu. (Fig. 5.6). Notice that T and T' have the same degree sequence. However, for any vertex $s \in (V(T) \setminus V(W)) \cup \{w\}$, $\mathrm{ecc}_{T'}(s) \geq \mathrm{ecc}_T(s)$ because $P_T(u,v)$ is a longest path in T. For any vertex $r \in V(W) - w$, we have

$$\mathrm{ecc}_{T'}(r) = d(r,u) + d(u,v) > d(u,v) \geq \mathrm{ecc}_T(r).$$

Thus $\mathrm{Ecc}(T') > \mathrm{Ecc}(T)$, which contradicts the extremality of T.

Since T is a caterpillar with internal vertices forming path $P = u_1u_2\dots u_k$, the eccentricity of any internal vertex is independent of the interval vertex degree assignments. For any $i \in [k]$ and leaf w adjacent to u_i,

$$\mathrm{ecc}_T(w) = \max\{k-i, i-1\} + 2.$$

If $\phi : \{u_i\}_{i=1}^k \to \{d_i\}_{i=1}^k$ is a one-to-one function, then when k is even,

$$\mathrm{Ecc}(T) = \sum_{i=1}^{k} \mathrm{ecc}_T(u_i) + (\phi(u_1) + \phi(u_k))(k+1) + (\phi(u_2) + \phi(u_{k-1}))(k) + \\ \dots + \left(\phi\left(u_{k/2}\right) + \phi\left(u_{(k+2)/2}\right)\right)(k/2+2).$$

In order to maximize the total eccentricity, for $i, j \in [k]$, if j is closer to $k/2$ than i, then we should have $\phi(u_i) \geq \phi(u_j)$. It is a greedy caterpillar which achieves this. The case when k is odd is similar. ∎

Greedy trees and level-greedy trees

In this subsection, each tree is rooted at a vertex. (While the root has no bearing on the total eccentricity, we use the added structure to direct our conversation.) The height of a vertex is the distance to the root and the tree's height, $h(T)$, is the maximum of all vertex heights. We start with some definitions.

Definition 5.12 (Schmuck, Wagner, and Wang (2012))**.** *In a rooted tree, the list of multisets L_i of degrees of vertices at height i, starting with L_0 containing the degree of the root vertex, is called the* level-degree sequence *of the rooted tree.*

Let $|L_i|$ be the number of entries in L_i. It is easy to see that a list of multisets is the level degree sequence of a rooted tree if and only if (1) the multiset $\bigcup_i L_i$ is a tree degree sequence, (2) $|L_0| = 1$, and (3) $\sum_{d \in L_0} d = |L_1|$, and for all $i \geq 1$, $\sum_{d \in L_i}(d-1) = |L_{i+1}|$.

In a rooted tree, the *down-degree* of the root is equal to its degree. The down degree of any other vertex is its degree minus one.

Definition 5.13 (Schmuck, Wagner, and Wang (2012))**.** *Given the level-degree sequence of a rooted tree, the* level-greedy rooted tree *for this level-degree sequence is built as follows: (1) For each $i \in [n]$, place $|L_i|$ vertices in level i and to each vertex, from left to right, assign a degree from L_i in non-increasing order. (2) For $i \in [n-1]$, from left to right, join the next vertex in L_i whose down-degree is d to the first d so far unconnected vertices on level L_{i+1}. Repeat for $i+1$.*

Definition 5.14 (Wang (2008))**.** *Given a tree degree sequence $(d_1, d_2, \ldots, d_n)$ in non-increasing order, the* greedy tree *for this degree sequence is the level-greedy tree for*

the level-degree sequence that has $L_0 = \{d_1\}$, $L_1 = \{d_2, \ldots, d_{d_1+1}\}$ *and for each* $i > 1$,

$$|L_i| = \sum_{d \in L_{i-1}} (d-1)$$

with every entry in L_i *at most as large as every entry in* L_{i-1}.

Fig. 5.7 shows a greedy tree with degree sequence $(4, 4, 4, 3, 3, 3, 3, 3, 3, 3, 2, 2, 1, \ldots, 1)$.

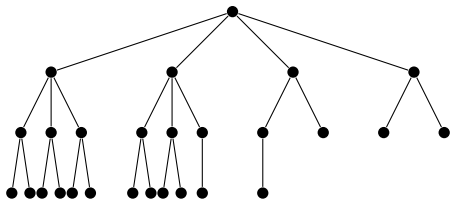


Figure 5.7 A greedy tree.

By definition, every greedy tree is level-greedy. However, Fig. 5.8 shows a level-greedy tree that is not greedy. It has level degree sequence:

$$\{\{3\}, \{5, 3, 2\}, \{3, 3, 3, 2, 2, 1, 1\}, \{2, 2, 1, 1, 1, 1, 1, 1\}, \{1, 1\}\}.$$

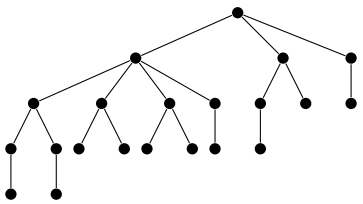


Figure 5.8 A level-greedy tree.

For a fixed degree sequence, greedy trees minimize the sum of distances (Schmuck, Wagner, and Wang 2012; Wang 2008; Zhang, Xiang, Xu, and Pan 2008) and maximize the number of subtrees (Andriantiana, Wagner, and Wang 2013; Zhang, Zhang, Gray, and Wang 2013). We will show that they also minimize $\mathrm{Ecc}(T)$ among trees with a given degree sequence.

Here we provide some set-up for the proofs of the next two theorems. See Fig. 5.9 for an illustration. Given a tree T rooted at v, let T_1 be the subtree, rooted at child

v_1 of v, containing some leaves of height $h := h(T)$. Let $h' := h(T - T_1)$. Then for any vertex $u \in V(T - T_1)$ and any $w \in V(T_1)$ with $h_T(u) = h_T(w) = j$, then

$$\mathrm{ecc}_T(u) = j + h, \tag{5.2}$$

$$\mathrm{ecc}_T(w) = \max\{j + h', \mathrm{ecc}_{T_1}(w)\} \leq j + h \tag{5.3}$$

where the first is only dependent on the height of T and the second depends only on h' and the structure of T_1.

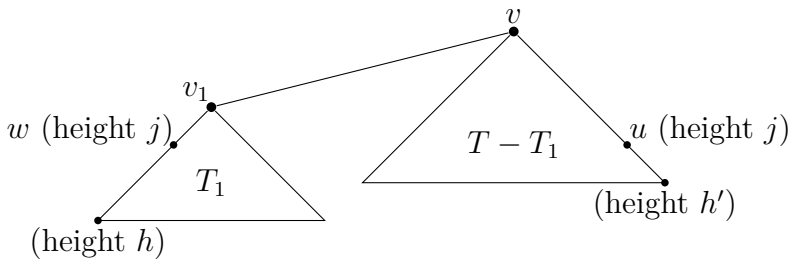


Figure 5.9 A tree rooted at v with daughter subtree, T_1, containing leaves of height h.

The following lemma implies that the level-greedy tree has the minimum total eccentricity among all rooted trees with a specified level-degree sequence.

Lemma 5.15. *Let ℓ be a non-negative integer. Among the trees with a given level-degree sequence, the level-greedy tree maximizes the number of vertices having eccentricity at most ℓ.*

Proof. We proceed by induction on the number of vertices. The base case with one vertex is trivial.

Fix $\ell > 0$. Let T be a rooted tree with the given level-degree sequence and the maximum number of vertices with eccentricity at most ℓ. (i.e. T is optimal.) For vertices $w \in T_1$ and $u \in T - T_1$, both of height j, suppose for contradiction that $\deg(u) > \deg(w)$. Create a new tree T' by moving $\deg(u) - \deg(w)$ children of u and

their descendants to adoptive parent w. This effectively switches the degrees of u and w while maintaining the level degree sequence.

While $\mathrm{ecc}_{T'}(u) = \mathrm{ecc}_T(u)$, notice that h' did not increase and neither did $\mathrm{ecc}_T(w)$ for $w \in V(T_1)$. Since $ecc_{T'}(w) \leq \max\{j + h', \mathrm{ecc}_{T_1}(w)\} = \mathrm{ecc}_T(w)$, if strict inequality holds, then we have contradicted the optimality of T. Otherwise, T' and T are both optimal trees. In this case, we can repeat this shifting of degrees for pairs of vertices of height 1, followed by pairs of vertices of height 2, and so on until we either meet a contradiction or construct an optimal tree in which $\deg(u) \leq \deg(w)$ for all $w \in T_1$ and $u \in T - T_1$ of the same height. Assume that our optimal T has this property.

Now we have a partition of the level-degree sequence for T into level-degree sequences for $T - T_1$. By the inductive hypothesis, we may assume that both T_1 and $T - T_1$ are level-greedy trees on their level-degree sequences. As a result, T is a level-greedy tree. ∎

The next theorem also yields a stronger result than merely minimizing total eccentricity among trees with a given degree sequence.

Theorem 5.16. *Fix $\ell \in \mathbb{Z}^{\geq 0}$. Among the trees with a given degree sequence, the greedy tree maximizes the number of vertices with eccentricity at most ℓ.*

Proof. Let T be a tree with the given degree sequence with the maximum number of vertices with eccentricity at most ℓ. (i.e. T is optimal.) Many times we will use the following claim: For two vertices u and v with $h(u) < \ell \leq h(v)$, it is preferable to assign degrees such that $\deg(u) \geq \deg(v)$ in order to maximize the number of vertices with height at most ℓ.

Find a longest path in T and root T at a center vertex v of that path. In $T - \{v\}$, let T_1 be the component with the leaf of greatest height. Let v_1 be the child of v in T_1. By our choice of the root, if h is the height of T_1, then the height of $T - T_1$ has height $h' \in \{h - 1, h\}$. Now for any $w \in V(T_1)$ with $h_T(w) = j$, we have

$\mathrm{ecc}_{T_1}(w) \leq (j-1)+(h-1) \leq j+h'-1$. In light of (5.3),

$$\mathrm{ecc}_T(w) = \max\{j+h', \mathrm{ecc}_{T_1}(w)\} = j+h'.$$

For $w, x \in V(T_1)$, if $h_T(w) < h_T(x)$ then, by our earlier claim, $\mathrm{ecc}_T(w) < \mathrm{ecc}_T(x)$ which implies $\deg(w) \geq \deg(x)$ in T because T maximizes the number of vertices with small eccentricities.

Vertices in $T - T_1$ with height j have eccentricity $j+h$ by (5.2). So for u, v in $V(T-T_1)$, when $h_T(u) < h_T(v)$, we can conclude $\deg(u) \geq \deg(v)$ in T.

These observations establish the fact that either the root of $T - T_1$ or the root of T_1 has the largest degree in T.

We now examine two cases based upon the value of h'. When $h = h'$, we have $\mathrm{ecc}_T(w) = j+h = \mathrm{ecc}_T(u)$ for any $w \in V(T_1)$, $u \in V(T-T_1)$ with $h_T(w) = h_T(u) = j$. Therefore, for $x, y \in V(T)$, if $h_T(x) < h_T(y)$, then $\deg(x) \geq \deg(y)$ in T. As an immediate consequence, the root of T has the largest degree.

When $h' = h-1$, we may assume that the root of T has the largest degree, for otherwise, we could reroot T at v_1 which would not change the vertex eccentricities or the difference between h and h'. Continuing in the setting with $h' = h-1$, for $w \in V(T_1)$ and $u, y \in V(T-T_1)$, if $h_T(w) = h_T(u)$, then $\mathrm{ecc}_T(w) = \mathrm{ecc}_T(u) - 1$. So $\deg(w) \geq \deg(u)$ in T. However, if $h_T(w) \geq h_T(y) + 1$, then $\mathrm{ecc}_T(w) \geq \mathrm{ecc}_T(y)$. So we may assume $\deg(w) \leq \deg(y)$ in T.

In both cases, we may assume that vertices of smaller height have larger degrees. Consequently, this determines the level degree sequence of T. In fact, this is the level degree sequence for the greedy tree. The previous lemma asserts that we can assume T is level-greedy. Therefore, T is the greedy tree. ∎

Remark 5.17. *Extremal trees for total eccentricity are not unique. In Theorem 5.16, we proved that the greedy tree had a stronger property. But in the proof, we can see that the greedy tree is not even unique in this regard.*

Greedy trees with different degree sequences

As a final remark on greedy trees, given a collection of degree sequences, we order the corresponding greedy trees by their total eccentricity. The following observations, similar to previous works on other indices, yields many extremal results as immediate corollaries. For an example of such applications see Zhang, Zhang, Gray, and Wang (2013).

Definition 5.18. *Given two non-increasing sequences in* $\mathbb{R}^n$, $\pi' = (d'_1, \cdots, d'_n)$ *and* $\pi'' = (d''_1, \cdots, d''_n)$, π'' *is said to majorize* π', *denoted* $\pi' \triangleleft \pi''$, *if for* $k \in [n-1]$

$$\sum_{i=0}^{k} d'_i \leq \sum_{i=0}^{k} d''_i \qquad \textit{and} \qquad \sum_{i=0}^{n} d'_i = \sum_{i=0}^{n} d''_i.$$

Lemma 5.19 (Wei (1982))**.** *Let* $\pi' = (d'_1, \cdots d'_n)$ *and* $\pi'' = (d''_1, \cdots, d''_n)$ *be two non-increasing tree degree sequences. If* $\pi' \triangleleft \pi''$, *then there exists a series of (non-increasing) tree degree sequences* $\pi^{(i)} = (d_1^{(i)}, \ldots, d_n^{(i)})$ *for* $1 \leq i \leq m$ *such that*

$$\pi' = \pi^{(1)} \triangleleft \pi^{(2)} \triangleleft \cdots \triangleleft \pi^{(m-1)} \triangleleft \pi^{(m)} = \pi''.$$

In addition, each $\pi^{(i)}$ *and* $\pi^{(i+1)}$ *differ at exactly two entries, say the* j *and* k *entries,* $j < k$ *where* $d_j^{(i+1)} = d_j^{(i)} + 1$ *and* $d_k^{(i+1)} = d_k^{(i)} - 1$.

Remark 5.20. *Lemma 5.19 is a more refined version of the original statement in Wei (1982). In this process, each entry stays positive and the degree sequences remain non-increasing. Thereby, each obtained sequence is a tree degree sequence that is non-increasing without rearrangement.*

Theorem 5.21. *Given two tree degree sequences* π' *and* π'' *such that* $\pi' \triangleleft \pi''$,

$$\mathrm{Ecc}(T^*_{\pi'}) \geq \mathrm{Ecc}(T^*_{\pi''})$$

where T^*_ν *is the greedy tree for degree sequence* ν.

Proof. According to Lemma 5.19, it suffices to compare the total eccentricity of two greedy trees whose degree sequences differ in two entries, each by exactly 1, i.e., assume

$$\pi' = (d'_1, \cdots d'_n) \triangleleft (d''_1, \cdots, d''_n) = \pi''$$

with $d''_j = d'_j + 1$, $d''_k = d'_k - 1$ for some $j < k$ and all other entries the same.

Let u and v be the vertices corresponding to d'_j and d'_k respectively and w be a child of v in $T^*_{\pi'}$ (Fig. 5.10). Construct $T_{\pi''}$ from $T^*_{\pi'}$ by removing the edge vw and adding edge uw. Note that $T_{\pi''}$ has degree sequence π'' and by Theorem 5.16

$$\mathrm{Ecc}(T^*_{\pi''}) \leq \mathrm{Ecc}(T_{\pi''}).$$

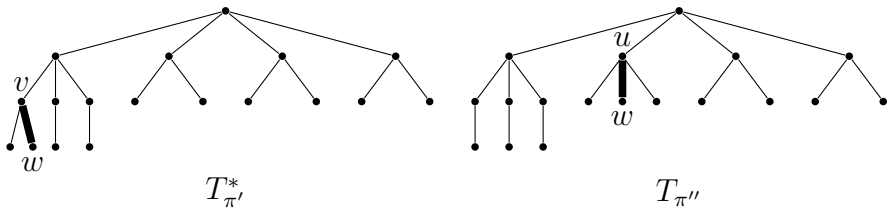


Figure 5.10 Creating $T_{\pi''}$ from $T_{\pi'}$ when
$\pi' = (4, 4, 3, 3, 3, 3, 2, 2, 1, \ldots, 1)$ and
$\pi'' = (4, 4, 4, 3, 3, 2, 2, 2, 1, \ldots, 1)$.

The height of any vertex in $T_{\pi''}$ is at most that of its counterpart in $T^*_{\pi'}$. An argument similar to that used in the proof of Lemma 5.15 shows

$$\mathrm{Ecc}(T_{\pi''}) \leq \mathrm{Ecc}(T^*_{\pi'}). \tag{5.4}$$

Hence $\mathrm{Ecc}(T^*_{\pi''}) \leq \mathrm{Ecc}(T_{\pi''}) \leq \mathrm{Ecc}(T^*_{\pi'})$. ∎

Remark 5.22. *As in the proof of the extremality of greedy trees, equality holds more often in* (5.4) *compared with its analogue for many other graph invariants. This also serves as some indication that* $\mathrm{Ecc}(T)$ *is not as strong of a graph invariant as compared to others in terms of characterizing the structures.*

By comparing greedy trees with different degree sequences, the extremality of trees with respect to minimizing Ecc(.) under various restrictions easily follows. Consider, for example, trees with a given number of vertices and exactly ℓ leaves. The degree sequence of such a tree has exactly ℓ of 1's, where the degree sequence $(\ell, 2, \ldots, 2, 1, \ldots, 1)$ majorizes all other possible degree sequences. The corresponding greedy tree is a "star-like" tree (a subdivision of star). Similarly, for trees with a given number of vertices and maximum degree k, the degree sequence $(k, k, \ldots, k, \ell, 1, \ldots 1)$ majorizes all other degree sequences with maximum degree k, where ℓ is the unique degree that is possibly between 1 and k. The corresponding greedy tree is called the "extended good k-ary" tree. See for instance, Bartlett, Krop, Magnant, Mutiso, and Wang (2014) or Zhang, Zhang, Gray, and Wang (2013) for details.

Chapter 6

On different "middle parts" of a tree

There are many different vertex attributes that have been placed on trees. We are interested in the eccentricity, distance, and number of subtrees. Each attribute naturally defines a set as the middle of the tree. For example, the study of the center and centroid (Definitions 6.1 and 6.2) can be traced back to Jordan (1869). We explore extremal problems regarding the distance between the vertices in the different middle parts.

6.1 Definitions and Characterizations

Below we define each attribute. The distance in the tree from u to v, denoted $d(u, v)$, is the number of edges on the unique connecting path $P(u, v)$.

Definition 6.1. *The* eccentricity *of a vertex v in a tree T is the largest distance that one can travel in T when starting at v. More specifically,*

$$ecc_T(v) = \max_{u \in V(T)} d(v, u).$$

The center *of T, denoted $C(T)$, is the set of vertices which have the minimum eccentricity among all vertices in the tree.*

Definition 6.2. *The* distance *of a vertex v, denoted $d(v)$, is the sum of distances from v to each other vertex in T,*

$$d(v) = \sum_{u \in V(T)} d(v, u).$$

The centroid *of T, denoted $CT(T)$, is the set of vertices which have the minimum distance among all vertices in the tree.*

A subtree of tree T is a connected subgraph which is induced on a set of vertices. We consider T to be a subtree of itself and a single vertex is also a subtree of T.

Definition 6.3. *As the name suggests, the number of subtrees of a vertex v, denoted $F_T(v)$, is the number of subtrees of T which contain v. The* subtree core *of a tree T, denoted by $Core(T)$, is the set of vertices that maximize the function $F_T(.)$ (Székely and Wang 2005).*

If H is a forest and v is a vertex in H, then $F_H(v)$ will be defined, as above, to be the number of subtrees of H which contain vertex v. In particular, all subtrees which are counted must be subtrees of the component of H which contains vertex v.

Jordan (1869) found that $C(T)$ consists of either one vertex or two adjacent vertices (see also Ex. 6.21a in Lovász (2007)). Given the vertices along any path of a tree, the sequence of $F_T(.)$ function values is strictly concave down (Székely and Wang (2005)), the sequence of $d(.)$ function values are strictly concave up (Ex. 6.22 in Lovász (2007); Entringer, Jackson, and Snyder (1976)), and the sequence of $ecc_T(.)$ function values are concave up (Ex. 6.21 in Lovász (2007)). Strict concavity immediately implies that the sets $CT(T)$ and $Core(T)$ either consist of one vertex or two adjacent vertices.

We are specifically interested in how the middle sets are located, relative to one another. It is well-known that $C(T)$ and $CT(T)$ can be far apart (Ex. 6.22c in Lovász (2007)), and that $Core(T)$ can differ from them (Székely and Wang 2005).

There are some natural questions that we will explore. How close to each other can they be? How far apart can they be spread? Must they lie on a common path? Can they appear in any ordering?

It is easy to find trees where $C(T)$, $CT(T)$, and $Core(T)$ coincide, like the star and paths of even length to name a few. It is more interesting to see how far apart these middle sets can be in a single tree.

As any two edges always lie on a common path, the vertices from any two of the sets $C(T)$, $CT(T)$, and $Core(T)$ always lie on a path. However, it is possible that the vertices from $C(T)$, $CT(T)$, and $Core(T)$ in the same tree T do not all lie on a common path. Figure 6.1 provides an example of this very situation.

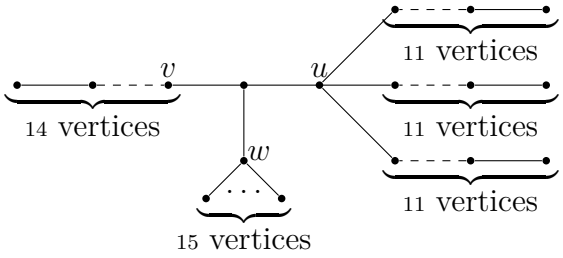


Figure 6.1 A tree with $v \in C(T)$, $u \in CT(T)$, $w \in Core(T)$ which do not lie on a common path.

On the other hand, when the vertices of $C(T)$, $CT(T)$, $Core(T)$ happen to lie on the same path, they can appear in any order. Figure 6.2 provides some illustrations.

Among the examples with different ordering of middle vertices, it is interesting to observe that vertices in $Core(T)$ generally have large degree; vertices in $C(T)$ generally have small degree; while vertices in $CT(T)$ behave somewhat between the previous two.

Here, we formalize some necessary and sufficient conditions for a vertex to be in a middle part. While not novel, these propositions and their proofs are included for completeness.

Proposition 6.4. *Let T be a tree with at least two vertices. A vertex v is in the center $C(T)$ if and only if there are two leaves, u and w, such that $P(v,u) \cap P(v,w) = \{v\}$, $d(v,u) = ecc_T(v)$, and $d(v,w) \geq ecc_T(v) - 1$.*

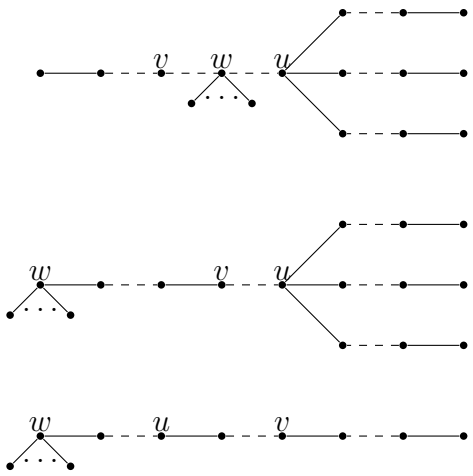


Figure 6.2 An example of trees with vertices $v \in C(T)$, $u \in CT(T)$, $w \in Core(T)$ on a common path, but in different orders.

Proof. Fix $v \in C(T)$. We consider two cases based upon the value of $ecc_T(v) =: x$. If $x = 1$, then T is a star with v in the center. In particular, if there are more than two vertices, any two leaves will serve the purpose. If T consists of only two vertices, they will both be leaves which meet the criteria.

Otherwise $x \geq 2$. By the definition of eccentricity, there is a leaf u such that $P(v, u)$ has x edges. Further, for any other leaf w, the length of $P(v, w)$ is at most x.

Let T' be the forest obtained from T by deleting the vertices of $P(v, u)$, except v. Let H be the connected component of T' which contains v. Suppose for contradiction that, for every $w \in V(H)$, $d_H(v, w) \leq x - 2$. Let v' be the unique neighbor of v on $P(v, u)$. Observe $d_T(v', u) = x-1$ and $d_T(v', w) \leq d_H(v, w)+1 = d_T(v, w)+1 \leq x-1$ for every $w \in V(H)$. For any leaf $a \in L(T \setminus H)$, $d_T(v', a) + 1 = d_T(v, a) \leq x$. Thus $ecc_T(v') = x - 1$. This contradicts the choice of v because $ecc_T(v') \leq ecc_T(v)$. Therefore there is at least one $w \in L(H)$ has $d_H(v, w) \in \{x - 1, x\}$.

On the other hand, if a vertex v has the property that there are leaves u, w with disjoint paths $P(u, v)$ and $P(v, w)$ with $d(v, u) = ecc_T(v)$ and $d(v, w) \geq ecc_T(v) - 1$, then the eccentricity of the vertices of $P(u, w)$ is at least that of v. All other vertices

have eccentricity at least one more than $ecc_T(v)$. Therefore $v \in C(T)$. ∎

Corollary 6.5. *If there are two leaves u, w such that $d(v,w) = d(v,u) = ecc_T(v)$, then $C(T) = \{v\}$. If no such w exists, then $|C(T)| = 2$ where the neighbor of v on $P(u,v)$ is also in the center.*

Next we give a characterization of the vertices in $CT(T)$.

Proposition 6.6. *Let T be a tree with at least two vertices. A vertex u is in the centroid $CT(T)$ if and only if for each neighbor v of u, we have*

$$n_{uv}(v) \leq n_{uv}(u)$$

where $n_{uv}(u)$ ($n_{uv}(v)$) denotes the number of vertices in the component containing u (v) in $T - uv$ which is the result after the deletion of edge uv from T.

Proof. For any two neighboring vertices u and v, it is easy to see

$$d(u) = d(v) + n_{uv}(v) - n_{uv}(u). \tag{6.1}$$

Therefore $d(u) \leq d(v)$ exactly when $n_{uv}(u) \geq n_{uv}(v)$. Indeed, along any path uvw, $n_{uv}(u) < n_{vw}(v)$ because when vw is a removed, the component containing v is a superset of the component containing u when the edge uv was removed. (The inequality is strict because the first set contains vertex v while the second does not.) Similarly, $n_{uv}(v) > n_{vw}(w)$. Whenever $d(u) \leq d(v)$, the inequalities established here imply

$$n_{vw}(w) - n_{vw}(v) < n_{uv}(v) - n_{uv}(u) \leq 0,$$

from which we can conclude $d(v) < d(w)$. Therefore, whenever u is a vertex such that $d(u) \leq d(v)$ for each of its neighbors v, then u is in the centroid. The converse of this statement holds by the definition of the centroid and making use of equation 6.1 which implies that $d(u) \leq d(v)$ if and only if $n_{uv}(v) \geq n_{uv}(u)$. ∎

Lastly, Proposition 6.7 gives a characterization of $Core(T)$.

Proposition 6.7. *A vertex u is in $Core(T)$ if and only if for each neighbor v of u, we have*

$$F_{T-uv}(u) \geq F_{T-uv}(v).$$

Proof. For any two adjacent vertices u and v, let $T_{uv}(u)$ denote the component of $T - uv$ which contains u. Each subtree H of T which contains v is characterized by two subtrees: the intersection of H with $T_{uv}(u)$ and the intersection of H with $T_{uv}(v)$. Note that the intersection with $T_{uv}(u)$ may be empty. Therefore

$$F_T(v) = F_{T-uv}(v)\left(F_{T-uv}(u) + 1\right) = F_{T-uv}(v)F_{T-uv}(u) + F_{T-uv}(v).$$

Likewise

$$F_T(u) = F_{T-uv}(u)F_{T-uv}(v) + F_{T-uv}(u).$$

Therefore, $F_T(u) \geq F_T(v)$ exactly when $F_{T-uv}(u) \geq F_{T-uv}(v)$. Now if $u \in Core(T)$, then $F_T(u) \geq F_T(v)$ for all $v \in V(T)$ and consequently $F_{T-uv}(u) \geq F_{T-uv}(v)$ for each neighbor v of u. This proves one direction of the proposition.

For any path uvw, notice that each subtree of $T - uv$ which contains u can be identified with a subtree of $T - vw$ which contains v. Just include the edge uv in the subtree. Therefore $F_{T-uv}(u) < F_{T-vw}(v)$. (The inequality is strict because $\{v\}$ is a tree in the second set which will not be identified with any tree in the first collection.) A similar argument shows $F_{T-vw}(w) < F_{T-uv}(v)$.

Now assuming $F_T(u) \geq F_T(v)$ which holds exactly when $F_{T-uv}(u) \geq F_{T-uv}(v)$, we determine

$$F_{T-vw}(w) < F_{T-uv}(v) \leq F_{T-uv}(u) < F_{T-vw}(v). \tag{6.2}$$

This is equivalent to $F_T(w) < F_T(v)$.

Now if every neighbor v of vertex u has the property $F_{T-uv}(u) \geq F_{T-uv}(v)$, then $F_T(u) \geq F_T(v)$. We have shown that this extends to the neighbors w of v such that $F_T(w) < F_T(v)$ and the argument continues out to every vertex in the graph implying that u is in $Core(T)$ since it has the largest number of subtrees of T containing it. ∎

6.2 Maximum distances between middle parts in general trees

Fix an arbitrary $n \in \mathbb{Z}^+$. Among all trees with n vertices, we determine the maximum distance that can be realized between vertices from different middle parts. We will see that the maximum distances are achieved by the structure named "comets."

Definition 6.8 (Barefoot, Entringer, and Székely (1997))**.** *An* r-comet *of order* n *is formed by attaching* $n - r$ *pendant vertices to one end vertex of a path on* r *vertices (Figure 6.3).*

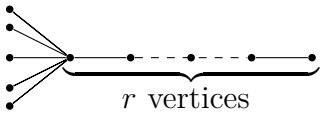


Figure 6.3 An r-comet of order n.

For vertex sets S, S' in a tree T, the quantity $\min\{d(u,v) : u \in S, v \in S'\}$ will be denoted $d(S, S')$.

6.2.1 Between center and centroid

Theorem 6.9. *Fix an arbitrary* $n \in \mathbb{Z}^+$*. For any tree* T *with* n *vertices,*

$$d(C(T), CT(T)) \leq \left\lfloor \frac{n-3}{4} \right\rfloor. \tag{6.3}$$

Proof. Fix a tree T on n vertices. Let $v \in C(T)$ and $u \in CT(T)$ such that the graph distance between u and v is precisely $d(C(T), CT(T))$. Therefore, no vertex on the path $P(u,v)$ other than u and v is in the center or the centroid of T.

Let $P(u,v)$ denote the path connecting u and v and let T_u denote the component containing u in $T - E(P(u,v))$. By Proposition 6.6,

$$|V(T_u)| > n - |V(T_u)|.$$

This implies

$$|V(T_u)| > \frac{n}{2}.$$

Let w be a leaf such that $P(v,w)$ and $P(u,v)$ are disjoint, except for v, and the length of $P(v,w)$ is maximum. Because $v \in C(T)$ and the neighbor of v on $P(u,v)$ is not in $C(T)$, Proposition 6.4 tells

$$d(v,w) = ecc_T(v).$$

Since u is not a leaf, it is easy to see that

$$d(u,v) \leq ecc_T(v) - 1.$$

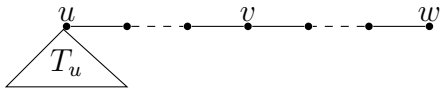


Figure 6.4 A tree T with $u \in CT(T)$, $v \in C(T)$, $w \in L(T)$, and all vertices not in T_u are on the path $P(u,w)$.

Therefore, we have

$$\begin{aligned} \frac{n}{2} &> n - |V(T_u)| && (6.4)\\ &\geq d(u,v) + d(v,w) && (6.5)\\ &\geq 2d(u,v) + 1. && (6.6) \end{aligned}$$

This implies

$$d(u,v) < \frac{n-2}{4}.$$

In particular, if $n = 4k + r$ with $r \in \{0,1,2\}$, then

$$k - \frac{1}{2} \leq \frac{4k + r - 2}{4} \leq k.$$

Since $d(u,v) < \frac{n-2}{4}$, when $n \equiv r \mod 4$ for $r \in \{0,1,2\}$, $d(u,v) \leq k-1$ where $k = \left\lfloor \frac{n}{4} \right\rfloor$.

When $n = 4k + 3$,

$$d(u, v) < \frac{n-2}{4} = \frac{4k+1}{4} = k + \frac{1}{4}.$$

As a result, $d(u, v) \leq k$. ∎

Proposition 6.10. *Let $k := \left\lfloor \frac{n}{4} \right\rfloor$. Equality holds in* (6.3) *exactly when n, T fall into one of the following categories:*

- *$n = 4k$ and T is the $2k$-comet.*
- *$n = 4k + 1$ or $n = 4k + 2$ and T is one of the following trees:*
  - *$2k$-comet*
  - *$2k$-comet on $n - 1$ vertices together with one vertex pendant to one of the internal vertices of $P(u, w)$*
  - *$P(u, w)$ has $2k+1$ vertices and T_u is a tree which is rooted at u, has height at most 2, and has $n - 2k - 1$ non-root vertices.*
- *$n = 4k + 3$ and T is a $(2k + 2)$-comet.*

Proof. If n is even, then $\frac{n}{2} > n - |V(T_u)|$ is equivalent to $\frac{n}{2} - 1 \geq n - |V(T_u)|$. Therefore

$$\frac{n}{2} - 1 \geq n - |V(T_u)| \geq d(u, v) + d(v, w) \geq 2d(u, v) + 1$$
$$d(u, v) \leq \frac{n}{4} - 1$$

When $n = 4k$, the trees with $d(u, v) = k - 1$ are precisely those trees in which all of the inequalities above are equalities. In other words, T_u (which includes vertex u) has $2k + 1$ vertices, all vertices not in T_u lie on the path $P(u, w)$ and all vertices not on $P(u, w)$ are pendant to u. This characterizes the $2k$-comet.

If $n = 4k+2$, then the above inequalities indicate $d(u, v) \leq k-\frac{1}{2}$. Because $d(u, v)$ is an integer, $d(u, v) \leq k - 1$. To analyze the extremal structures, first observe that

$\frac{n}{2} - 1$ is even while $2d(u,v) + 1$ is odd. Therefore not all inequalities can be tight. Indeed, exactly one will be strict. If the first inequality is not strict, then we have a $2k$-comet on n vertices. If the second inequality is the one which is not strict, we have a $(2k+1)$-comet on $n-1$ vertices with one vertex pendant to one of the internal vertices on the path $P(u,w)$. If the last inequality is not strict, we have a tree rooted at u on $2k+1$ non-root vertices and has height at most 2, together with a disjoint path $P(u,w)$ with $2k+1$ vertices.

When n is odd, we have

$$\frac{n-1}{2} \geq n - |V(T_u)| \geq d(u,v) + d(v,w) \geq 2d(u,v) + 1$$
$$d(u,v) \leq \frac{n-3}{4}.$$

When $n = 4k+3$, then $d(u,v) = k$ will hold precisely when all inequalities are tight. This happens when T is a $(2k+2)$-comet.

When $n = 4k+1$, $\frac{n-1}{2}$ is even while $2d(u,v)+1$ is odd. Therefore exactly one of the inequalities must not be strict. If the first inequality is not strict, then we have a $(2k+1)$-comet. If the second inequality is not strict, then we have a $2k$-comet on $n-1$ vertices with one vertex pendant to one of the internal vertices of $P(u,w)$. If the last inequality is not strict, then T_u is a tree which is rooted a u, has height at most, and has $2k$ non-root vertices will the rest of the tree is just the path $P(u,w)$ with $2k+1$ vertices. ∎

6.2.2 Between centroid and subtree core

Next we turn our attention to the centroid and the subtree core.

Theorem 6.11. *Let T be a tree with $n > 8$ vertices. If $n \geq 2^{\lceil \log_2 n \rceil - 1} + \lceil \log_2 n \rceil$, then*

$$d(CT(T), Core(T)) \leq \left\lfloor \frac{n-1}{2} \right\rfloor - \lfloor \log_2 n \rfloor - 1$$

with equality for the $(n - \lfloor \log_2 n \rfloor - 1)$ *- comet. Otherwise*

$$d(CT(T), Core(T)) \leq \left\lfloor \frac{n-1}{2} \right\rfloor - \lfloor \log_2 n \rfloor .$$

with equality holding for the $(n - \lfloor \log_2 n \rfloor)$*-comet.*

Proof. Let $u \in CT(T)$ and $v \in Core(T)$ in a tree T with $|V(T)| = n$ and the graph distance between u and v is precisely $d(CT(T), Core(T))$. Let $P(u, v)$ denote the path connecting u and v and let T_u, T_v denote the components containing u, v respectively in $T - E(P(u, v))$. Set $x := |V(T_u)|$ and $y := |V(T_v)|$. Ultimately, we desire an upper bound for $d(u, v)$ together with an extremal example. Observe

$$d(u, v) \leq n - x - y + 1.$$

Thus we desire lower bounds for x and y.

Since $u \in CT(T)$ and the neighbor of u on $P(u, v)$ is not in $CT(T)$, Proposition 6.6 implies

$$x > n - x$$
$$x > \frac{n}{2}.$$

More precisely, $x \geq \left\lceil \frac{n+1}{2} \right\rceil$.

Next we bound y. Because $v \in Core(T)$ and the neighbor of v on $P(u, v)$ is not in $Core(T)$, Proposition 6.7 gives

$$F_{T_v}(v) > F_{T - T_v}(w)$$

where w is the unique neighbor of v on $P(u, v)$. See Figure 6.5 for an illustration of how these pieces interact.

Further note that every subtree in T_v which contains v can be uniquely identified by the set of its vertices, excluding v. Thus,

$$F_{T_v}(v) \leq 2^{y-1}.$$

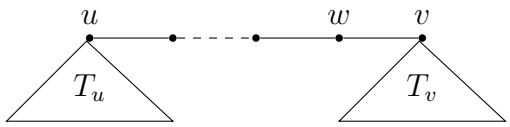


Figure 6.5 A representation of tree T for the proof of Theorem 6.11 with path $P(u, v)$, T_u, T_v, and w labeled.

Note that equality holds if and only if every subset of vertices induces a tree which is the case exactly when T_v is a star centered at v. On the other hand, since $T - T_v$ is a tree, subtrees can be created from $T - T_v$ by iteratively deleting a leaf, which is not v, of $T - T_v$. The result is

$$F_{T-T_v}(w) \geq n - y.$$

Equality holds here if and only if $T - T_v$ is a path with w as an end vertex.

Putting these inequalities together, we see that when $v \in Core(T)$,

$$2^{y-1} > n - y.$$

As a linear and an exponential equation in y, if $2^{y_0-1} = n - y_0$, then for all $y > y_0$, $2^{y-1} > n - y$. Therefore, we proceed by solving $2^{y_0-1} = n - y_0$. Note that $y_0 > 0$.

$$\begin{aligned} 2^{y_0-1} &= n - y_0 \\ y_0 - 1 &= \log_2(n - y_0) \\ y_0 &= \log_2(n - y_0) + 1 \\ &< \log_2(n) + 1. \end{aligned}$$

Using the equation $y_0 = \log_2(n - y_0) + 1$ and substituting into itself, we find

$$\begin{aligned} y_0 &= \log_2(n - \log_2(n - y_0) - 1) + 1 \\ &\geq \log_2(n - \log_2(n) - 1) + 1 \\ &> \log_2(n). \end{aligned}$$

The last inequality will be proven later for $n > 8$.

As a result, we have the bounds

$$\log_2(n) < y_0 < \log_2(n) + 1.$$

Further, if $y_0 < \lfloor \log_2 n \rfloor + 1$, then $2^y - 1 > n - y$ precisely when $y \geq \lfloor \log_2 n \rfloor + 1$. However, if $y_0 \geq \lfloor \log_2 n \rfloor + 1$ then $2^y - 1 > n - y$ precisely when $y \geq \lfloor \log_2 n \rfloor + 2$. Note that these are the two condition in the theorem statement.

Now we show $\log_2(n - \log_2 n - 1) > \log_2 n - 1$ for $n > 8$. First observe $\log_2 n < \frac{n}{2} - 1$ for $n > 8$. Therefore

$$\begin{aligned}
\log_2 n &< \frac{n}{2} - 1 \\
\frac{n}{2} &< n - \log_2 n - 1 \\
\log_2\left(\frac{n}{2}\right) &< \log_2(n - \log_2 n - 1) \\
\log_2 n - 1 &< \log_2(n - \log_2 n - 1).
\end{aligned}$$

When $n > 8$, our bounds for integers x and y give

$$\begin{aligned}
d(u, v) &\leq n - x - y + 1 \\
&\leq n - \left\lceil \frac{n+1}{2} \right\rceil - \lfloor \log_2 n \rfloor \\
&= \left\lfloor \frac{n-1}{2} \right\rfloor - \lfloor \log_2 n \rfloor .
\end{aligned}$$

As mentioned earlier, this can be strengthen to $d(u, v) \leq \left\lfloor \frac{n-1}{2} \right\rfloor - \lfloor \log_2 n \rfloor - 1$ if $y_0 \geq \lfloor \log_2 n \rfloor + 1$. However, this will only happen if $2^{\lfloor \log_2 n \rfloor} \leq n - \lfloor \log_2 n \rfloor - 1$ as stated in the theorem.

As for extremal trees, equality will hold in the upper bound for $d(u, v)$ if T_u has exactly $\left\lceil \frac{n+1}{2} \right\rceil$ vertices and T_v is a star while $T - T_v$ is a path. This describes the C-comet where $C = n - \lfloor \log_2 n \rfloor$ or in the case where $n \geq 2^{\lfloor \log_2 n \rfloor} + \lceil \log_2 n \rceil + 1$, $C = n - \lfloor \log_2 n \rfloor - 1$. ∎

6.2.3 Between subtree core and center

Theorem 6.12. *For any tree T on $n > 8$ vertices, if $n \geq 2^{\lceil \log_2 n \rceil - 1} + \lceil \log_2 n \rceil$ then*

$$d(C(T), Core(T)) \leq \left\lfloor \frac{1}{2}(n - \lfloor \log_2 n \rfloor - 2) \right\rfloor$$

which is tight for the K-comet with $K = n - \lfloor \log_2 n \rfloor + 1$. Otherwise

$$d(C(T), Core(T)) \leq \left\lfloor \frac{1}{2}(n - \lfloor \log_2 n \rfloor - 1) \right\rfloor$$

which is tight for the K-comet with $K = n - \lfloor \log_2 n \rfloor$.

Proof. Let $u \in Core(T)$ and $v \in C(T)$ in a tree T with $|V(T)| = n$ and the graph distance between u and v is precisely $d(C(T), Core(T))$. Use T_u (respectively T_v) to denote the component containing u (v) in $T - E(P(u, v))$ and let $y = |V(T_u)|$.

Because $v \in C(T)$ and the neighbor of v on $P(u, v)$ is not in $C(T)$, there is a leaf w in T_v with $d(v, w) = ecc_T(v)$. As argued in the proof of Theorem 6.9,

$$d(u, v) \leq ecc_T(v) - 1 < d(v, w),$$

$$2d(u, v) + 1 \leq d(u, v) + d(v, w) \leq n - y.$$

Note that these inequalities are tight for the $(n - y + 1)$-comet.

Because $u \in Core(T)$, we can conclude, as in the proof of Theorem 6.11,

$$2^{y-1} > n - y.$$

Consequently,

$$y \geq \lfloor \log_2 n \rfloor .$$

Combining inequalities, we obtain the bound in the theorem statement:

$$d(u, v) \leq \left\lfloor \frac{1}{2}(n - y - 1) \right\rfloor \leq \left\lfloor \frac{1}{2}(n - \lfloor \log_2 n \rfloor - 1) \right\rfloor .$$

Recall from Theorem 6.11 that if $n \geq 2^{\lceil \log_2 n \rceil - 1} + \lceil \log_2 n \rceil$, then

$$y \geq \lfloor \log_2 n \rfloor + 1$$

and consequently we obtain the better bound

$$d(u,v) \leq \left\lfloor \frac{1}{2}(n-y-1) \right\rfloor \leq \left\lfloor \frac{1}{2}(n-\lfloor \log_2 n \rfloor - 2) \right\rfloor.$$

■

6.3 Trees with degree restrictions

In this section, we narrow our sights to classes of trees that follow certain degree restrictions. First we fix a degree sequence and see some results for trees which realize this degree sequence. Next, we consider only binary trees in which all non-root vertices have degree 1 or 3 while the root may have degree 2 or 3. Lastly, we fix integers n, k and consider classes of tree with n vertices and maximum degree k.

For trees with a maximum degree condition, we obtain results about the distance between their "middle parts." In order to prove these, we first obtain results about the maximum or minimum number of root-containing subtrees a tree with a specified degree sequences can have. Note that among trees with n vertices, it is the path, rooted at one end, which minimizes the number of root-containing subtrees and the star, rooted at the center vertex, which maximizes the number of root-containing subtrees.

6.3.1 Trees with a given degree sequence

In Chapter 5, Definition 5.14, we defined the greedy tree, an extremal structure explored in many previous studies.

Fix a degree sequence for a tree and distinguish a single value in this sequence which will be the degree of the root. Similar to the greedy tree, we define the *rooted greedy tree*.

Definition 6.13. *Let $\overline{d} = (d_1, d_2, \ldots, d_n)$ be a tree degree sequence in non-increasing order with degree d_i identified as the root degree. Let $(d'_1, d'_2, \ldots, d'_{n-1})$ be the degree*

sequence $\overline{d}$ in non-increasing order with d_i removed. The rooted greedy tree for this degree sequence is the level-greedy tree for the level-degree sequence that has $L_0 = \{d_i\}$, $L_1 = \{d'_1, \ldots, d'_{d_i}\}$ and for each $j > 1$,

$$|L_j| = \sum_{d \in L_{j-1}} (d-1)$$

with the largest element in L_j less than or equal to the smallest element in L_{j-1}.

Figure 6.6 shows a rooted greedy tree with root degree 2 and degree sequence

$$(4, 3, 3, 3, 3, 2, 1, \ldots, 1). \tag{6.7}$$

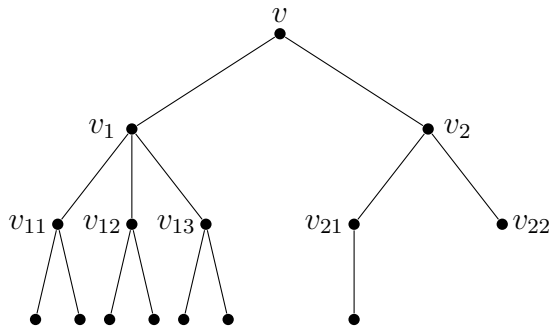


Figure 6.6 A rooted greedy tree with degree sequence (6.7) and root degree 2.

Among trees with given degree sequence, greedy trees are extremal with respect to many graph invariants. For example, the following result is for root-containing subtrees.

Theorem 6.14 (Andriantiana, Wagner, and Wang (2013))**.** *Fix a degree sequence $\overline{d}$ and a positive integer k. Among rooted trees with degree sequence $\overline{d}$, the number of subtrees which contain the root and have exactly k vertices is maximized by the greedy tree. Consequently the greedy tree maximizes the total number of root-containing subtrees.*

Fix a degree sequence, distinguish one value in the sequence as the root degree, and fix a positive integer k'. Among rooted trees with this degree sequence and the

specified root degree, the number of subtrees containing the root and having k' vertices is maximized by the rooted greedy tree. Consequently, the rooted greedy tree maximizes the total number of root-containing subtrees.

6.3.2 Binary trees

The study of binary trees is well motivated from its applications in phylogeny. Székely and Wang (2005) studied the number of subtrees of a binary tree. They found that the extremal structures are *good trees*, *rgood trees*, and *caterpillars*. In our terms, a good binary tree is a greedy tree with root degree 3 and degree sequence

$$\{3, \ldots, 3, 1, \ldots, 1\}$$

and an rgood binary tree is a rooted greedy tree with root degree 2 and degree sequence

$$\{3, \ldots, 3, 2, 1, \ldots, 1\}.$$

A binary caterpillar consists of a path P with pendant vertices that make the degree of each internal vertex 3.

Their results for the number of subtrees are as follows:

Theorem 6.15 (Székely and Wang (2007))**.** *Among all binary trees with n leaves, where every non-leaf vertex has degree 3, the good binary tree minimizes the number of subtrees.*

Theorem 6.16 (Székely and Wang (2005))**.** *Among binary trees with n leaves, the binary caterpillar on n leaves minimizes the number of subtrees.*

As an immediate consequence of Theorem 6.14, we obtain the following results for root-containing subtrees.

Corollary 6.17. *Among all binary trees, the good binary tree has the maximum number of root-containing subtrees.*

Corollary 6.18. *Fix $n \in \mathbb{Z}^{\geq 0}$. Among all rooted binary trees with n vertices, the rgood binary tree is the unique tree which maximizes the number of root-containing subtrees.*

For binary trees, we can examine the distance between vertices of different middle parts in much the same way that we did in Section 6.2. While the exact calculations are quite messy, we conjecture the following result.

Conjecture 6.19. *Among binary trees of order n, the tree T, formed from identifying the root of an rgood binary tree with a vertex of maximum eccentricity in a binary caterpillar (Figure 6.7), maximizes the distance between*

1. *the closest pair $u \in CT(T)$ and $v \in C(T)$,*
2. *the closest pair $u \in Core(T)$ and $v \in CT(T)$,*
3. *the closest pair $u \in Core(T)$ and $v \in C(T)$.*

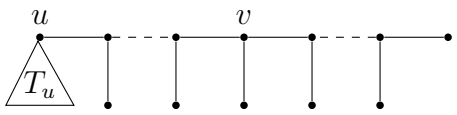


Figure 6.7 An extremal binary tree which is conjectured to maximize the distances $d(CT(T), C(T))$, $d(Core(T), CT(T))$, and $d(Core(T), C(T))$ for u and v as in Conjecture 6.19. The tree T_u is an rgood binary tree.

6.3.3 Trees with bounded maximum degree

In this section, we turn our focus to trees on n vertices, all of which have degree at most k.

We previously defined good binary trees and rgood binary trees. In general, for each positive integer k, a good tree is a greedy tree with degree sequence

$$(k, k, \ldots, k, 1, 1, \ldots 1)$$

while the rgood trees are rooted greedy trees with root degree $k-1$ and degree sequence

$$(k, k, \ldots, k, k-1, 1, 1, \ldots 1).$$

For any fixed k, these trees only exist for certain values of n. Therefore, we extend their definitions as follows so that we can create similar trees for any $n > k$.

For positive integers n, k $(n > k)$, a tree with order n and maximum degree k is called an *extended good tree* if it is a greedy tree with degree sequence

$$(k, k, \ldots, k, s, 1, \ldots, 1)$$

for some $1 \leq s < k$ (Figure 6.8). Notice that the degree sequence is determined by n and k. The value s is the remainder when we divide $n-1$ by k. If $n-1 = qk+s$ then there will be q vertices of degree k, one of degree s, and the rest will be leaves.

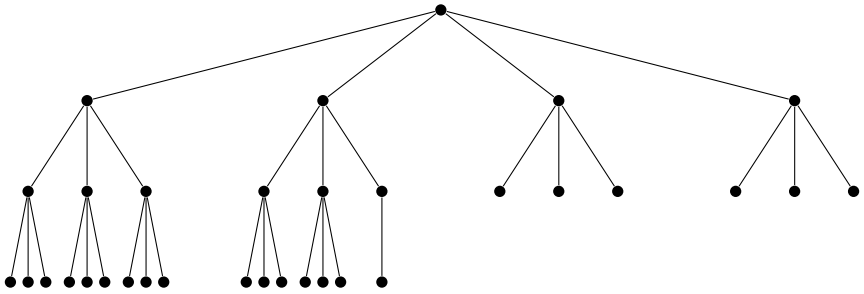


Figure 6.8 An extended good tree with 33 vertices and maximum degree 4.

Similarly, for positive integers n, k, the *extended rgood tree* with order n and maximum degree k, is a rooted greedy tree with root degree $k-1$ and degree sequence

$$(k, k, \ldots, k, k-1, s, 1, \ldots, 1)$$

for some $1 \leq s < k$ (Figure 6.9). The value of s will be the remainder when dividing n by k.

Among all rooted trees with n vertices, maximum degree k, and root degree $\rho \leq k-1$, we seek the one with the maximum number of root-containing subtrees. We call such a tree *optimal*.

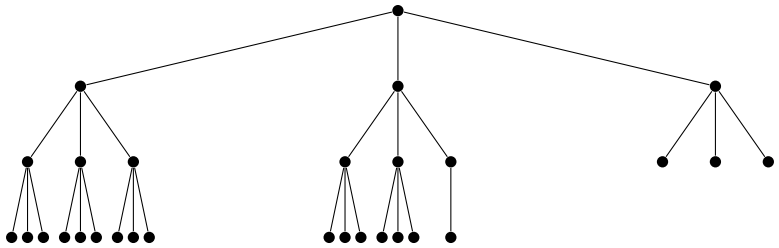


Figure 6.9 An extended rgood tree with 29 vertices and maximum degree 4.

Theorem 6.20. *Among all rooted trees with n vertices, maximum degree k, and root degree $\rho \leq k-1$, the extended rgood tree maximizes the number of root-containing subtrees.*

To provide a proof for this theorem, we first establish two lemmas.

Lemma 6.21. *An optimal tree with $n \geq k$ must have root degree $k-1$.*

Proof. For contradiction, suppose T is an optimal tree with root r having degree $\rho \leq k-2$. Since $n \geq k$, there exists a child u of r that is not a leaf. Let v be a child of u and T_v be the subtree induced by v and its descendants.

Define $T' := T - \{uv\} + \{rv\}$. Every root-containing subtree in T can be uniquely identified by its list of vertices. It is easy to see that each list forms a root-containing subtree in T'. However, T' also has root-containing subtrees which contain v and not u. These do not appear in T. Therefore T' has more root-containing subtrees than T. This contradicts our choice of T. ∎

Definition 6.22. *For degree sequences $\pi = (d_0, \cdots, d_{n-1})$ and $\pi' = (d'_0, \cdots, d'_{n-1})$, π' majorizes π, denoted $\pi \triangleleft \pi'$, if for each $k \in \{0, \cdots, n-2\}$,*

$$\sum_{i=0}^{k} d_i \leq \sum_{i=0}^{k} d'_i \qquad \text{and} \qquad \sum_{i=0}^{n-1} d_i = \sum_{i=0}^{n-1} d'_i.$$

The following is a simpler analogue of Theorem 11 of Andriantiana, Wagner, and Wang (2013). We skip the routine argument.

Lemma 6.23. *Let T and T' be rooted greedy trees on n vertices with root degree $k-1$. If T has degree sequence π and T' has degree sequence π' where $\pi \triangleleft \pi'$, then T' has more root-containing subtrees than T.*

In the search for an optimal tree, Lemma 6.21 implies that it is sufficient to restrict our attention to trees with root degree $k-1$. Because we are considering only degree sequences on n vertices with maximum degree k, it is easy to see that the degree sequence of the extended rgood tree majorizes all other such degree sequences. Thus, Lemma 6.23 then implies that the extended rgood tree for order n and maximum degree k as stated in Theorem 6.20.

Remark 6.24. *Note that, among all rooted trees of given order, root degree at most $k-1$, and maximum degree k:*

- *the extended rgood tree minimizes the height;*
- *the path (rooted at one end) minimizes the number of root-containing subtrees and maximizes the height.*

6.3.4 In trees with a given maximum degree k

Fix $n, k \in \mathbb{Z}^{>0}$. Similar to the binary tree case, we restrict our attention to classes of trees which have order n and maximum degree k. Then we look for trees in this class which maximize the distance between different "middle parts." Our findings are detailed in this section.

Theorem 6.25. *For fixed $n, k \in \mathbb{Z}^{>0}$, any tree T with order n and maximum degree k has*

$$d(CT(T), C(T)) \leq \frac{n - \left\lceil \frac{n+1}{2} \right\rceil - h_u}{2}$$

where

$$h_u = \left\lceil \frac{\ln\left(\lceil \frac{n+1}{2} \rceil (k-2) + 1 \right)}{\ln(k-1)} \right\rceil - 1.$$

This inequality is tight for the tree formed from an extended rgood tree and a path by identifying the root of the extended rgood tree with one end of a path.

Proof. Select $u \in CT(T)$ and $v \in C(T)$ such that $d(u,v) = d(CT(T), C(T))$. Let T_u and T_v name the components containing u and v respectively in $T - E(P(u,v))$.

Counting the vertices in T, we obtain the inequality

$$d(u,v) \leq n - |V(T_u)| - |V(T_v)| + 1. \tag{6.8}$$

Because $u \in CT(T)$, Proposition 6.6 implies the following for the vertex in the centroid which is closer to the center vertices:

$$\begin{aligned} |V(T_u)| &> n - |V(T_u)|, \\ |V(T_u)| &\geq \left\lceil \frac{n+1}{2} \right\rceil. \end{aligned}$$

Set h_u and h_v equal to the heights of T_u and T_v respectively. Because $v \in C(T)$, Proposition 6.4 implies the following:

$$\begin{aligned} d(u,v) + h_u &\leq h_v, \\ d(u,v) &\leq h_v - h_u \leq |V(T_v)| - 1 - h_u. \end{aligned} \tag{6.9}$$

The upper bound for $d(u,v)$ is tight when $h_v = |V(T_v)| - 1$, which happens exactly when T_v is a path, and h_u is minimum.

By Remark 6.24, the minimum h_u is achieved when T_u is the extended rgood tree. Since $|V(T_u)| \geq \left\lceil \frac{n+1}{2} \right\rceil$ and the maximum degree is k, we can determine the height of an extended rgood tree with these conditions. The extended rgood tree with maximum degree h and height h has at most $\sum_{i=0}^{h}(k-1)^i$ vertices. For T_u with n vertices, the height h will be the smallest value which satisfies

$$\begin{aligned} |V(T_u)| &\leq \sum_{i=0}^{h}(k-1)^i \\ &= \frac{1-(k-1)^{h+1}}{1-(k-1)} \\ &= \frac{(k-1)^{h+1}-1}{k-2}. \end{aligned}$$

Next we solve for h.

$$\begin{aligned}
|V(T_u)|(k-2) &\leq (k-1)^{h+1} - 1 \\
|V(T_u)|(k-2) + 1 &\leq (k-1)^{h+1} \\
\ln\left(|V(T_u)|(k-2)+1\right) &\leq (h+1)\ln(k-1) \\
\frac{\ln\left(|V(T_u)|(k-2)+1\right)}{\ln(k-1)} - 1 &\leq h.
\end{aligned}$$

Since h is the smallest value that satisfies the above inequality and $|V(T_u)| = \left\lceil \frac{n+1}{2} \right\rceil$, we can conclude

$$h = \left\lceil \frac{\ln\left(\left\lceil \frac{n+1}{2} \right\rceil (k-2) + 1\right)}{\ln(k-1)} \right\rceil - 1.$$

Without knowing $|V(T_v)|$ exactly, we can add (6.8) and (6.9) and solve for $d(u,v)$ to obtain the desired upper bound for $d(u,v)$:

$$\begin{aligned}
2d(u,v) &\leq n - |V(T_u)| - h_u \\
d(u,v) &\leq \frac{1}{2}\left(n - |V(T_u)| - h_u\right) \\
&\leq \frac{1}{2}\left(n - \left\lceil \frac{n+1}{2} \right\rceil - h_u\right) \\
&\leq \left\lfloor \frac{1}{2}\left(n - \left\lceil \frac{n+1}{2} \right\rceil - h\right) \right\rfloor.
\end{aligned}$$

■

Theorem 6.26. *For fixed $n, k \in \mathbb{Z}^{>0}$, any tree T with order n and maximum degree k has*

$$d(Core(T), CT(T)) \leq n - n' - \left\lceil \frac{n+1}{2} \right\rceil + 1$$

where n' is the minimum order of an extended rgood tree T_u with maximum degree k such that $F_{T_u}(u) \geq n - |V(T_u)|$. This inequality is tight for the tree formed from an extended rgood tree and a path by identifying the root of the extended rgood tree with one end of a path.

Proof. Let $u \in Core(T)$ and $v \in CT(T)$ such that $d(u,v) = d(Core(T), CT(T))$. Define T_u and T_v to be the components of $T - E(P(u,v))$ containing u and v respectively. Let h_u and h_v be the heights of T_u and T_v respectively.

By Proposition 6.6, $v \in CT(T)$ and its neighbor on $P(u,v)$ is not in the centroid precisely when

$$|V(T_v)| \geq n - |V(T_v)| + 1,$$
$$|V(T_v)| \geq \left\lceil \frac{n+1}{2} \right\rceil.$$

By Proposition 6.7, $u \in Core(T)$ and its neighbor w on $P(u,v)$ is not in the subtree core precisely when

$$F_{T_u}(u) \geq 1 + F_{T-T_u}(w) \geq d(u,v) + F_{T_v}(v) \geq d(u,v) + |V(T_v)|,$$
$$d(u,v) \leq F_{T_u}(u) - F_{T_v}(v) \leq F_{T_u}(u) - |V(T_v)|. \tag{6.10}$$

The last inequality is tight if T_v is a path.

Counting the vertices in T, we see

$$n \geq d(u,v) + |V(T_u)| + |V(T_v)| - 1,$$
$$d(u,v) \leq n - |V(T_u)| - |V(T_v)| + 1 \leq n - \left\lceil \frac{n+1}{2} \right\rceil - n' + 1.$$

where n' is the minimum number of vertices in a tree T_u with maximum degree k such that $F_{T_u}(u) \geq d(u,v) + |V(T_v)| = n - |V(T_u)|$ as in (6.10). Note that $F_{T_u}(u)$ is maximized by the extended rgood tree, giving the extremal tree in the theorem statement. ∎

Theorem 6.27. *For fixed $n, k \in \mathbb{Z}^{>0}$, any tree T with order n and maximum degree k has*

$$d(Core(T), C(T)) \leq n - n' - \left\lfloor \frac{1}{2}(n - n' + h') \right\rfloor$$

where $h' = \left\lceil \frac{\ln(n'(k-2)+1)}{\ln(k-1)} \right\rceil - 1$ and n' is the minimum number of vertices in the extended rgood tree T_u with maximum degree k such that $F_{T_u}(u) \geq n - |V(T_u)|$. This inequality

is tight for the tree formed from an extended rgood tree and a path by identifying the root of the extended rgood tree with one end of a path.

Proof. Let $u \in Core(T)$ and $v \in C(T)$ such that $d(u,v) = d(Core(T), C(T))$. Define T_u and T_v to be the components of $T - E(P(u,v))$ containing u and v respectively. Let h_u and h_v be the heights of T_u and T_v respectively.

Because $u \in Core(T)$ and its neighbor on $P(u,v)$ is not in the subtree core, as in (6.10), Proposition 6.7 gives

$$d(u,v) \leq F_{T_u}(u) - |V(T_v)| \tag{6.11}$$

which is tight when T_v is a path.

Because $v \in C(T)$ and its neighbor on $P(u,v)$ is not in the center, as in the proof of Theorem 6.25, Proposition 6.4 gives

$$d(u,v) \leq h_v - h_u \leq |V(T_v)| - h_u - 1.$$

As in (6.9), this is also tight when T_v is a path.

Adding these two inequalities together we obtain the following bound.

$$d(u,v) \leq \frac{1}{2}\left(F_{T_u}(u) - h_u - 1\right).$$

The upper bound is maximum when $F_{T_u}(u)$ large and h_u is small which is optimized when T_u is the extended rgood tree.

If n' is the number of vertices in T_u, then because $v \in C(T)$ and T_v is a path, then $ecc_T(v)$ is at least half of the diameter of T which translates to

$$|V(T_v)| \geq \frac{1}{2}(n - n' + h_u).$$

Any tree on n' vertices with maximum degree at most k will have height at least the height of the corresponding extended rgood tree. As determined in the proof of Theorem 6.25,

$$h_u \geq \left\lceil \frac{\ln\left(n'(k-2)+1\right)}{\ln(k-1)} \right\rceil - 1 := h'.$$

In conclusion,

$$d(u,v) \leq n - |V(T_u)| - |V(T_v)| \leq n - n' - \left\lfloor \frac{1}{2}(n - n' + h') \right\rfloor.$$

Further, this upper bound is maximized when n' is minimized. However, n' must still satisfying the condition $F_{T_u}(u) \geq d(u,v) + |V(T_v)| = n - |V(T_u)|$ from (6.11). ∎

6.4 Different "middle parts" in trees with a given diameter D

In this section, for fixed $n, D \in \mathbb{Z}^{>0}$, consider classes of trees with n vertices and diameter at most D. The next two propositions follow from exactly the same arguments as those of Section 6.2, we skip the proofs.

Proposition 6.28. *For fixed D and large n, every tree T of order n and diameter at most D satisfies*

$$d(C(T), CT(T)) \leq \left\lfloor \frac{D-2}{2} \right\rfloor,$$

which is achieved by a D-comet.

Proposition 6.29. *For fixed D and large n, every tree T of order n and diameter at most D satisfies*

$$d(C(T), Core(T)) \leq \left\lfloor \frac{D-2}{2} \right\rfloor,$$

which is achieved by a D-comet.

The argument for $d(CT(T), Core(T))$ is more complex. Fix D and n. Among all trees with diameter at most D and order n, fix a tree T which realizes the maximum value for $d(CT(T), Core(T))$.

Select vertices $u \in Core(T)$ and $v \in CT(T)$ such that the graph distance between u and v is precisely $d(CT(T), Core(T))$. In $T - E(P(u,v))$, let T_u name the component containing u while T_v is the component containing v. Consider u to be the root of T_u and v to be the root of T_v.

Let w be the neighbor of u on $P(u,v)$. Because $u \in Core(T)$, Proposition 6.7 implies

$$F_{T_u}(u) < F_{T-T_u}(w).$$

Because $v \in CT(T)$ and its neighbor on $P(u,v)$ is not in $CT(T)$, Proposition 6.6 implies

$$|V(T_v)| > n - |V(T_v)|.$$

Suppose T_u is not a star. Create a new tree T' from T by replacing T_u with a star T'_u which is rooted at u and has the same order as T_u. Using the convention that T'_u and T'_v are the components containing u and v respectively in $T' - E(P(u,v))$, we see that T'_v is the same tree as T_v. First observe that

$$F_{T'_u}(u) \geq F_{T_u}(u) > F_{T-T_u}(w) = F_{T'-T'_u}(w)$$

which implies $w \notin Core(T')$ by Proposition 6.7. Further,

$$|V(T'_v)| = |V(T_v)| > n - |V(T_v)| = n - |V(T'_v)|$$

which implies the neighbor of v on $P(u,v)$ is not in the centroid of T' by Proposition 6.6. Therefore

$$d(Core(T'), CT(T')) \geq d_{T'}(u,v) = d_T(u,v) = d(Core(T), CT(T)).$$

By the choice of T, $d(Core(T'), CT(T')) = d(Core(T), CT(T))$. So T' is a tree with diameter at most D and order n which maximizes $d(Core(T), CT(T))$.

Now consider the structure of T'_v in T'. Say T'_v has x vertices and height h. Suppose T'_v does not minimize the number of subtrees containing v for its height and order. Let T''_v be a tree rooted at v with height at most h and order x which minimizes $F_{T''_v}(v)$. Define T'' to be the tree created from T' by replacing T'_v with T''_v. Observe that

$$F_{T''_u}(u) = F_{T'_u}(u) > F_{T'-T'_u}(w) > F_{T''-T''_u}(w)$$

which implies $w \notin Core(T'')$ by Proposition 6.7. Further, for T''_v being the component of $T'' - E(P(u,v))$ which contains v,

$$|V(T''_v)| = |V(T'_v)| > n - |V(T'_v)| = n - |V(T''_v)|.$$

This implies, by Proposition 6.6, that the neighbor of v on $P(u,v)$ in T'' is not in $CT(T'')$ and

$$d(Core(T''), CT(T'')) \geq d_{T''}(u,v) = d_T(u,v) = d(Core(T), CT(T)).$$

However, T was chosen as a tree which maximizes the distance between the subtree core and the centroid. Therefore $d(Core(T''), CT(T'')) = d(Core(T), CT(T))$. Therefore, T'' is also a tree with diameter at most D and order n that maximizes the distance between the subtree core and the centroid.

Remark 6.30. *Fix $n, D \in \mathbb{Z}^{>0}$. Among all trees with diameter at most D and order n, there is a tree T, with T_u a star rooted at u and T_v a tree which minimizes the number of subtrees containing v for its height and order, which maximizes $d(Core(T), CT(T))$. This structure T is drawn in Figure 6.4.*

In Section 6.5, we take a closer look at the structure of T_v, a tree which minimizes the number of subtrees containing v for its height and order. While we determine many necessary properties of T_v, characterizing the exact structure is still an open problem.

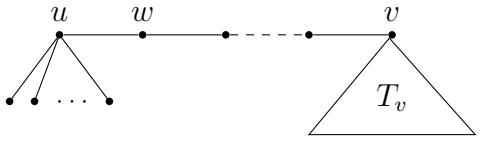


Figure 6.10 The structure of a tree T with diameter D and order n which maximize $d(Core(T), CT(T))$. Here, $u \in Core(T)$, $v \in CT(T)$, and T_v minimizes the number of subtrees containing v for its order and height.

For any $n, h \in \mathbb{Z}^{>0}$, this section is devoted to characterizing the rooted trees with n vertices and height at most h which have the minimum number of root-containing subtrees. We call these trees *optimal*.

To standardize some notation, we restrict our attention to trees T which are rooted at root ρ, have order n and height h unless mentioned otherwise. Note that $h(T) = ecc_T(r)$. The degree of a vertex v will be denoted $deg(v)$. It is necessary to have $h + 1 \leq n$ to guarantee that the tree will be realizable.

For any $v \in V(T)$, let $T(v)$ denote the subtree induced by v and all of its descendants. We will view $T(v)$ as a tree rooted at v. For each neighbor v_i of ρ, set $T_i := T(v_i)$. Here we present several observations regarding the characteristics of an optimal tree.

Lemma 6.31. *In any optimal tree T, for any $v \in V(T)$, $T(v)$ minimizes the number of root-containing subtrees among all rooted trees of the same order and height at most $h - h_T(v)$.*

Proof. Let T be an optimal tree. Suppose, for contradiction, that there is a vertex v for which $T(v)$ does not satisfy the lemma. In other words, there is a tree $T'(v)$, which is rooted at v, has the same order as $T(v)$, and has

$$h(T'(v)) \leq h - h_T(v) \text{ and } F_{T'(v)}(v) < F_{T(v)}(v).$$

Let T' be the tree obtained from T by replacing $T(v)$ with $T'(v)$. Then T and T' have the same number of subtrees containing ρ but not v. Define $T^* := T - (T(v) - \{v\})$ and let ρ and v. Because T and T' only differ in the descendants of v, we have

$$F_T(\rho) - F_{T'}(\rho) = F_{T(v)}(v)F_{T^*}(\rho, v) - F_{T'(v)}(v)F_{T^*}(\rho, v) > 0,$$

a contradiction to the optimality of T. ∎

Lemma 6.32. *The height of any leaf in an optimal tree is h.*

Proof. If $n = h+1$, it is straightforward to see that the path rooted at one end is the optimal tree. In the case when $n > h+1$, some vertex must have at least 2 children. Suppose, for contradiction, that there is a leaf $v \in V(T)$ whose height is less than h. Let x be the closest ancestor (possibly the root) of v that has at least two children. Let y be a child of x that is not on $P(x, v)$ and z be the child of x on $P(x, v)$.

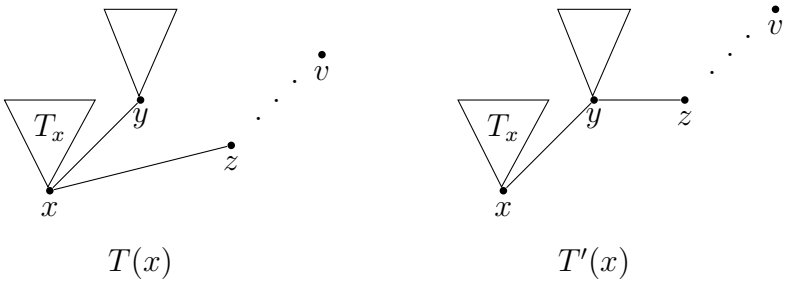


Figure 6.11 Trees $T(x)$ and $T'(x)$ from Lemma 6.32

Let T_x be the component containing x in $T - xy - xz$ and consider the tree

$$T'(x) := T(x) - xz + yz$$

depicted in Figure 6.11. Note that $T'(x)$ has the same order as $T(x)$ and has height no more than $h - h_T(x)$ because the height of v in T is less than h.

Counting the number of subtrees containing x in each tree, we obtain the following equalities:

$$\begin{aligned} F_{T(x)}(x) &= F_{T_x}(x)(1 + d_{T(x)}(x, v))(1 + F_{T(y)}(y)), \\ F_{T'(x)}(x) &= F_{T_x}(x)\left[1 + (1 + d_{T(x)}(x, v))F_{T(y)}(y)\right]. \end{aligned}$$

Together, these imply

$$F_{T(x)}(x) - F_{T'(x)}(x) = d_{T(x)}(x, v)F_{T_x}(x) > 0.$$

Since T was optimal, this contradicts Lemma 6.31. ∎

Lemma 6.33. *Every optimal tree has one of the following two properties:*

- *All non-root vertices have degree at most 3.*
- *All non-root vertices of height less than $h-1$ have degree at most 3. For any vertex v of height $h-1$, $deg(v) \leq 4$. Further, if $deg(v) = 4$, then the parent of v must have degree 2 or be the root.*

Proof. As before, this proof proceeds by contradiction. Let x be a non-root vertex in an optimal tree T with degree at least 4. Say y, z, and w are three children of x and let u be the parent of x. Denote by T_u and T_x the components containing u and x respectively in $T - ux - xy - xz - xw$. Without loss of generality, assume

$$F_{T(w)}(w) = \max\{F_{T(y)}(y), F_{T(z)}(z), F_{T(w)}(w)\}.$$

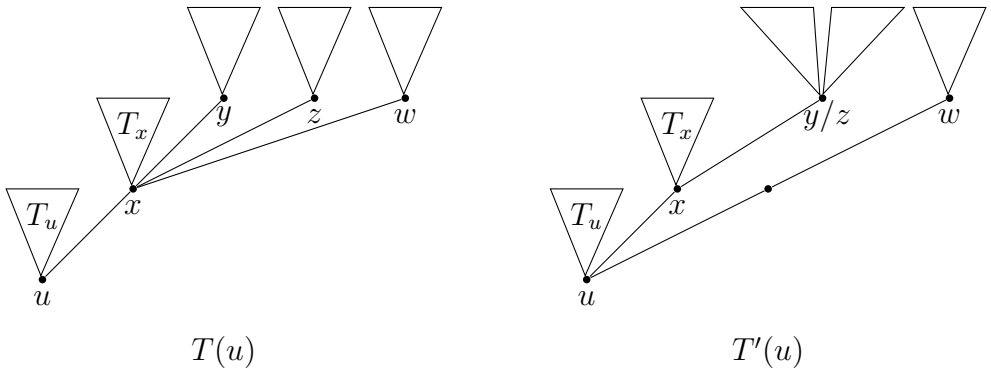


Figure 6.12 Trees $T(u)$ and $T'(u)$ in the proof of Lemma 6.33.

Now consider the tree $T'(u)$ obtained from $T(u)$ by removing the edges xz and xw, inserting a path of length 2 between u and w, while identifying the vertices y and z (Figure 6.12). Note that $T'(u)$ has the same height and order as $T(u)$. Counting the number of subtrees containing u in each, we find

$$\begin{aligned} F_{T(u)}(u) &= F_{T_u}(u)\left[1 + F_{T_x}(x)\left(1 + F_{T(y)}(y)\right)\left(1 + F_{T(z)}(z)\right)\left(1 + F_{T(w)}(w)\right)\right], \\ F_{T'(u)}(u) &= F_{T_u}(u)\left(2 + F_{T(w)}(w)\right)\left[1 + F_{T_x}(x)\left(1 + F_{T(y)}(y)F_{T(z)}(z)\right)\right]. \end{aligned}$$

Together, these imply the following

$$\begin{aligned} F_{T(u)}(u) - F_{T'(u)}(u) = F_{T_u}(u) \Big[& F_{T_x}(x) F_{T(y)}(y) \Big(F_{T(w)}(w) - F_{T(z)}(z) \Big) \\ & + F_{T_x}(x) \Big(F_{T(y)}(y) - 1 \Big) \\ & + \Big(F_{T(w)}(w) + 1 \Big) \Big(F_{T_x}(x) F_{T(z)}(z) - 1 \Big) \Big] \\ \geq 0. & \end{aligned} \tag{6.12}$$

Because T is an optimal tree, $T(u)$ is an optimal tree by Lemma 6.31. Therefore (6.12) must be equality. Note that for any tree H and vertex $a \in V(H)$, $F_H(v) \geq 1$ because the subtree containing only the vertex v will be counted. Therefore, equality holds in (6.12) exactly when $F_{T_x}(x) = F_{T(y)}(y) = F_{T(z)}(z) = F_{T(w)}(w) = 1$, or equivalently, $deg(x) = 4$ and y, z, w are all leaves so x has height $h-1$ in T. Create T' from T by replacing $T(u)$ with $T'(u)$. Because (6.12) is equality, $F_{T(u)}(u) = F_{T'(u)}(u)$. Therefore T' is also an optimal tree.

In T', $deg_{T'}(x) = 3$ but $deg_{T'}(u) = deg_T(u) + 1$. Observe u has height $h-2$ in T'. If u is not the root of T' and $deg_{T'}(u) \geq 4$, then we can repeat the argument for optimal tree T' and vertex u having degree at least 4. Because the height of u is $h-2$, we will find a contradiction in the step which parallels (6.12). Therefore $deg_{T'}(u) \leq 3$ which implies $deg_T(u) \leq 2$. Since u is not the root of T, we can conclude $deg_T(u) = 2$ as stated in the theorem. ∎

In the proof of Lemma 6.33, in the case where $deg_T(x) = 4$, we created another optimal tree T' where $deg_{T'}(x) = 3$ and no other degree 4 vertices where created. Hence, if an optimal tree has multiple degree 4 vertices of height $h-1$, we can repeat this procedure to obtain an optimal T' with all vertices of degree at most 3. This proves the following observation.

Observation 6.34. *There is an optimal tree in which all non-root vertices have degree at most 3.*

We now shift our attention to the structures of T_i for $1 \leq i \leq k$.

Lemma 6.35. *In an optimal tree T, each subtree $T_i \cup \{\rho\}$ falls into one of the following three categories:*

- *There is at most one non-root vertex with degree 3.*
- *All non-root vertices of height at most $h-3$ have degree 2, the vertex of height $h-2$ has degree 3, and exactly one of its children has degree 3.*
- *All non-root vertices of height at most $h-2$ have degree 2 and the vertex of height $h-1$ has degree 4.*

Proof. We prove this in two pieces, considering the alternatives from Lemma 6.33 separately. We start with the optimal trees in which all vertices have degree at most 3.

For contradiction, suppose there exists a $T_i \cup \{\rho\}$ with at least two non-root vertices of degree 3. Let v be a degree 3 vertex of greatest height in T_i and let u, w be the two children of v. Let z be the closest ancestor of v such that $deg_{T_i}(z) = 3$, z has parent x, and z has child $y \notin V(P(z, v))$. Let ℓ_1 denote the distance from v to a leaf in T_i and ℓ_2 the length of $P(v, z)$. Let T_x denote the component containing x in $T(x) - xz$ (Figure 6.13).

Create a new tree $T'(x)$ from $T(x)$ by removing the edges vw and zy, inserting a length 2 path between x and y, and identifying u and w (Figure 6.13). Note that $T'(x)$ has the same height and order as $T(x)$. The number of subtrees containing x in each is

$$F_{T(x)}(x) = F_{T_x}(x)\left[1 + (1 + F_{T(y)}(y))[\ell_2 + (\ell_1 + 1)^2]\right],$$
$$F_{T'(x)}(x) = F_{T_x}(x)(2 + F_{T(y)}(y))\left(\ell_2 + 2 + \ell_1^2\right).$$

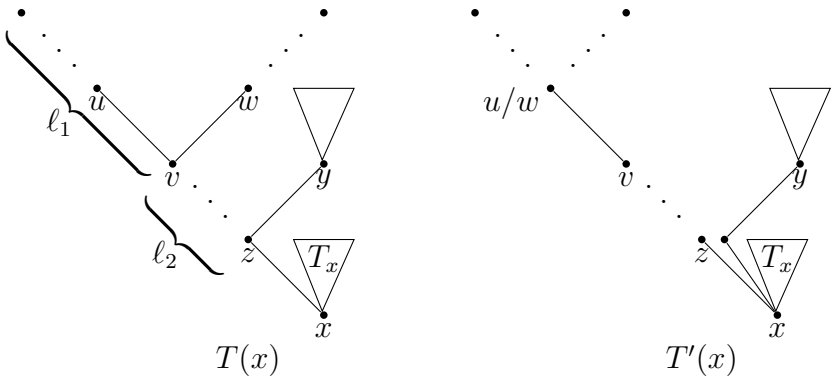


Figure 6.13 Transforming $T(x)$ into $T'(x)$ when $deg_T(v) = 3$ in the proof of Lemma 6.35.

By Lemma 6.32, the height of each leaf in T is h, hence $V(T(y)) \geq \ell_1 + \ell_2$. Now we have

$$\begin{aligned} F_{T(x)}(x) - F_{T'(x)}(x) &= F_{T_x}(x)\left[(1 + F_{T(y)}(y))(2\ell_1 - 1) - (\ell_1^2 + \ell_2 + 1)\right] \\ &\geq F_{T_x}(x)\left[(1 + \ell_1 + \ell_2)(2\ell_1 - 1) - (\ell_1^2 + \ell_2 + 1)\right] \qquad (6.13) \\ &= F_{T_x}(x)(\ell_1^2 + 2\ell_1\ell_2 + \ell_1 - 2\ell_2 - 2) \\ &\geq 0. \qquad (6.14) \end{aligned}$$

When either (6.13) or (6.14) is strict inequality, we have a contradiction to the optimality of T. Equality holds exactly when $\ell_1 = \ell_2 = 1$ and $|V(T(y))| = \ell_1 + \ell_2$. In other words, $T(y)$ is a single path on two vertices with y having height $h - 1$. Since T' (constructed from T by replacing $T(x)$ with $T'(x)$) is an optimal tree, if x is not the root, $deg_{T'}(x) \leq 3$ since x has height $h - 3$. Therefore $deg_T(x) \leq 2$ as described in the second property of the lemma.

If T falls into the second category listed in Lemma 6.33, then consider a subtree T_i with a vertex v of degree 4 at height $h - 1$. We will show that all other non-root vertices in $T_i \cup \{\rho\}$ must have degree 2. Suppose to the contrary that v has an ancestor z of degree 3. (In this way, we are able to simultaneously handle the case when there are two vertices of degree 4 in $T_i \cup \{\rho\}$ because they would have to share a common

ancestor of degree 3.) Label the vertices as before with s being the third child of v (Figure 6.14).

Create $T'(x)$ by altering $T(x)$ in a manner similar to that described above. Define

$$T'(x) = T(x) - wv - yz + xw + wy$$

as shown in Figure 6.14.

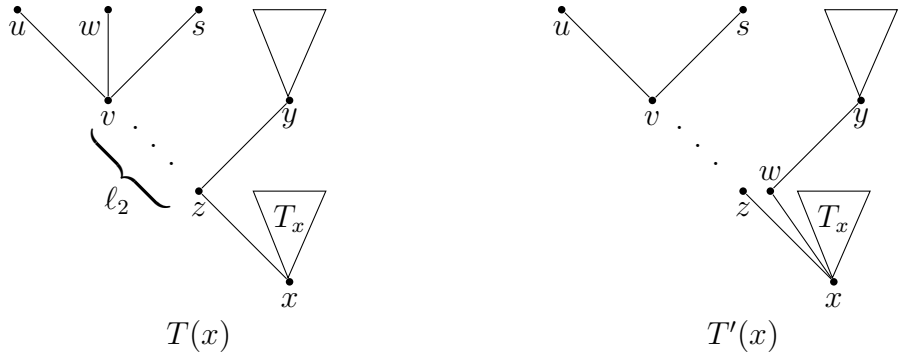


Figure 6.14 Transforming $T(x)$ into $T'(x)$ when v has degree 4 in the proof of Lemma 6.35.

Let ℓ_2 be the distance from z to v in $T(x)$. Because all leaves have height h, $F_{T(y)}(y) \geq \ell_2 + 1$ which is tight when $T(y)$ is a path. Now if we calculate $F_{T(x)}(x)$ and $F_{T'(x)}(x)$ exactly and take their difference, we find

$$\begin{aligned}
F_{T(x)}(x) &= F_{T_x}(x)\left(1 + (1 + F_{T(y)}(y))(\ell_2 + 8)\right) \\
F_{T'(x)}(x) &= F_{T_x}(x)(2 + F_{T(y)}(y))(\ell_2 + 5) \\
F_{T(x)}(x) - F_{T'(x)}(x) &= F_{T_x}(x)\left(3F_{T(y)}(y) - \ell_2 - 1\right) \\
&\geq F_{T_x}(x)\left(3(\ell_2 + 1) - \ell_2 - 1\right) \\
&= F_{T_x}(x)\left(2\ell_2 + 2\right) \\
&> 0.
\end{aligned}$$

This contradicts our choice of T. Thus $T_i \cup \{\rho\}$ can have at most one vertex of degree 4 and all other non-root vertices must have degree 2 as described in the third property of the lemma. ∎

Once again, it is useful to note that the optimal trees described in the second two properties of Lemma 6.35, the proof described T' analogues which have the same number of root-containing subtrees and yet fall under the first property description in Lemma 6.35. This gives the following observation.

Observation 6.36. *There is an optimal tree with each $T_i \cup \{\rho\}$ having at most one non-root vertex of degree 3.*

Define the *f-split* to be the tree rooted at v_1 with $h+f$ vertices, constructed from paths $P_1 = (v_1, v_2, \ldots, v_h)$, and $P_2 = (u_1, u_2, \ldots, u_f)$ by adding the edge $u_1 v_{h-f}$. We also define the 0-split to be merely a path on h vertices which is rooted at one end. By Observation 6.36, there is an optimal tree so that for each T_i there is $0 \leq k_i \leq h-1$ such that T_i is a k_i-split. First let us state a structural observation that will minimize some notation.

Observation 6.37. *In an optimal tree T, the number of root-containing subtrees in a k_i-split together with root ρ is*

$$s_h(k_i) := h + k_i^2 + k_i + 1.$$

This definition also makes sense for the 0-split, which has h root-containing subtrees, and the h-split, with h^2 subtrees that contain the root.

Lemma 6.38. *Among the T_i subtrees in an optimal tree, at most two of them can be 0-splits.*

Proof. Suppose, for contradiction, that T_i, T_j and T_k are each 0-splits in an optimal tree. Consider $S := T_i \cup T_j \cup T_k \cup \{\rho\}$. Create S' from S by replacing T_i with a 1-split, T_j with an $(h-1)$-split and deleting T_k (Figure 6.15).

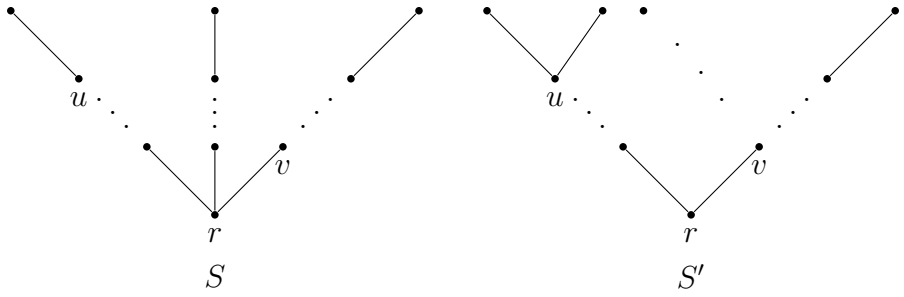


Figure 6.15 Trees S and S' from the proof of Lemma 6.38

The difference in the number of subtrees is

$$\begin{aligned} F_S(\rho) - F_{S'}(\rho) &= (s_h(0))^3 - s_h(1)s_h(h-1) \\ &= (h+1)^3 - (h+3)(2h + (h-1)^2) \\ &= 2(h-1) \\ &> 0. \end{aligned}$$

This contradicts the optimality of T because the tree obtained from T by replacing S with S' has fewer root-containing subtrees than T. ∎

Lemma 6.39. *If some T_i is a 0-split, then for each $j \neq i$, T_j is either a 0-split or a 1-split.*

Proof. Suppose instead that T_i is a 0-split and T_j is a k_j-split where $1 < k_j \leq h-1$. Let S be the tree induced by T_i, T_j and r. Construct S' from S by replacing T_i with a 1-split and replacing T_j with a $(k_j - 1)$-split (Figure 6.16).

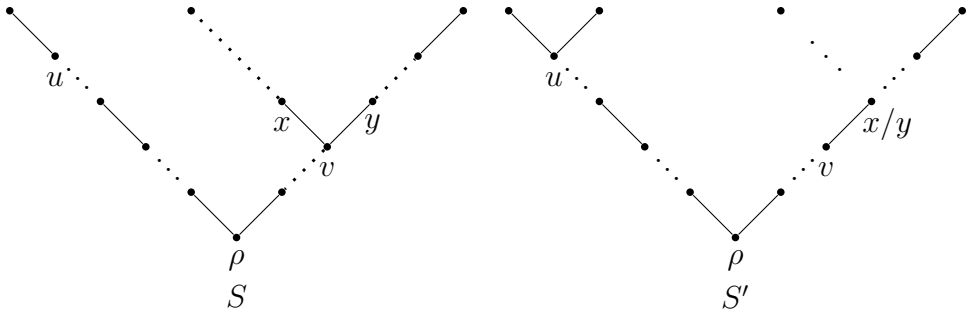


Figure 6.16 Trees S and S' from the proof of Lemma 6.39.

Note that S' has the same height and order as S, and

$$\begin{aligned}
F_S(\rho) - F_{S'}(\rho) &= s_h(0)s_h(k_j) - s_h(1)s_h(k_j - 1) \\
&= (h+1)\left[h + k_j^2 + k_j + 1\right] - (h+3)\left[h + (k_j-1)^2 + k_j\right] \\
&= (h+1)\left[h + k_j^2 + k_j + 1\right] - (h+3)\left[h + k_j^2 - k_j + 1\right] \\
&= 2k_j(h+1) - 2\left[h + k_j^2 - k_j + 1\right] \\
&= 2\left[k_j h - k_j^2 + k_j - h + k_j - 1\right] \\
&= 2\left[(k_j - 1)(h - k_j) + (k_j - 1)\right] \\
&> 0. \qquad \text{(for } k_j > 1\text{)}
\end{aligned}$$

This contradicts the optimality of T because the tree obtaining from T by replacing S with S' has fewer root-containing subtrees than T. ∎

Lemma 6.40. *A rooted tree T is not optimal if for any T_i (k_i-split) and T_j (k_j-split), we have $k_i(1 + k_j) > h + 1$ for $1 \le k_i \le k_j \le h - 1$.*

Proof. Define T_2 be the subtree of T which consists of the root ρ together with T_i and T_j.

Construct T_2' from T_2 by replacing T_i with a $(k_i - 1)$-split and replacing T_j with a $(k_j + 1)$-split. This construction is well-defined because $1 \le k_i$ and $k_j \le h - 1$.

It is easy to see that T_2' has the same height and order as T_2. We have

$$F_{T_2}(\rho) = s_h(k_i)s_h(k_j) \qquad \text{and} \qquad F_{T_2'}(\rho) = s_h(k_i - 1)s_h(k_j + 1).$$

Since $k_i \leq k_j$ and $k_i(1+k_j) > h+1$, we have

$$F_{T_2}(\rho) - F_{T_2'}(\rho) = -2(k_i - k_j - 1)(k_i + k_i k_j - h - 1) > 0,$$

which contradicts the optimality of T. ∎

By reversing the roles of k_i and k_j in the previous lemma, we obtain the following corollary.

Corollary 6.41. *A rooted tree T is not optimal if for any T_i, which is a k_i-split, and T_j, which is a k_j-split, we have $k_j(1+k_i) < h+1$ and $k_i < k_j - 1$.*

Corollary 6.42. *Fix an optimal tree T in which each T_i is a k_i-split with $k_i \geq k_{i+1}$. If $k_1 > \sqrt{h+\frac{5}{4}} - \frac{1}{2}$, then $k_i < \sqrt{h+\frac{5}{4}} - \frac{1}{2}$ for each $i \geq 2$.*

Proof. Let T be an optimal tree, as described in the corollary, with $k_1 > \sqrt{h+\frac{5}{4}} - \frac{1}{2}$. For contradiction, suppose $k_2 \geq \sqrt{h+\frac{5}{4}} - \frac{1}{2}$. Observe

$$\begin{aligned} k_2(k_1+1) &> \left(\sqrt{h+\frac{5}{4}} - \frac{1}{2}\right)\left(\sqrt{h+\frac{5}{4}} + \frac{1}{2}\right) \\ &= h + \frac{5}{4} - \frac{1}{4} \\ &= h+1. \end{aligned}$$

However, this contradicts the statement of Lemma 6.40. Therefore the corollary holds. ∎

Corollary 6.43. *Fix an optimal tree T in which each T_i is a k_i-split. For any pair $\{k_i, k_j\}$ with $k_i, k_j \leq \sqrt{h+\frac{5}{4}} - \frac{1}{2}$, we can conclude $|k_i - k_j| \leq 1$.*

Proof. Suppose $k_i, k_j \leq \sqrt{h+\frac{5}{4}} - \frac{1}{2}$ with $k_j \geq k_i + 2$. Observe

$$\begin{aligned} k_j(1+k_i) &\leq k_j(k_j-1) \\ &\leq \left(\sqrt{h+\frac{5}{4}} - \frac{1}{2}\right)\left(\sqrt{h+\frac{5}{4}} - \frac{3}{2}\right) \\ &= h + \frac{5}{4} - 2\sqrt{h+\frac{5}{4}} + \frac{3}{4} \\ &= h + 2 - 2\sqrt{h+\frac{5}{4}} \\ &< h+1. \end{aligned}$$

This contradicts Corollary 6.41, finishing the proof. ∎

Question 6.44. *Let root ρ have $deg(\rho) = r$ and let T_i be a k_i-split where $0 \leq k_i < h$ for each $i \in [r]$. If T is an optimal tree, then*

$$\begin{cases} \sum_{i=1}^{r} k_i + hr + 1 = n, \\ k_i + k_j \leq h, \qquad\qquad \text{for } 1 \leq i, j \leq r \text{ and } i \neq j. \end{cases}$$

Our goal is to minimize the expression

$$\prod_{i=1}^{k} (h + k_i^2 + k_i + 1),$$

obtained from the number of root-containing subtrees of an optimal tree illustrated in Figure 6.17.

While we do not yet have a complete characterization of optimal trees, we have many necessary properties.

Lemma 6.45. *Suppose $k_1 \geq k_2 \geq \ldots \geq k_k$. If $k_1 > \left\lceil \sqrt{h+\frac{5}{4}} - \frac{1}{2} \right\rceil$, then for each $i > 1$,*

$$\frac{h+1}{k_1} - 1 \leq k_i \leq \frac{h+1}{k_1+1}. \tag{6.15}$$

In particular, $k_2 = k_3 = \ldots = k_k = \left\lfloor \frac{h+1}{k_1+1} \right\rfloor$ provided $\left\lfloor \frac{h+1}{k_1+1} \right\rfloor \geq \frac{h+1}{k_1} - 1$.

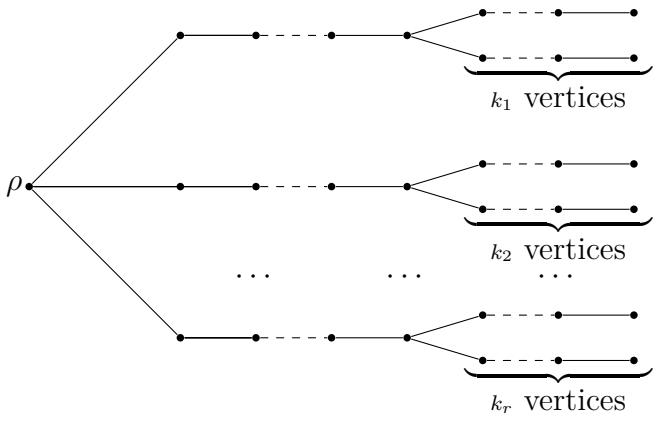


Figure 6.17 The structure of a tree T with height h and order n which minimizes the number of root-containing subtrees.

Proof. Since $k_1 > \sqrt{h+\frac{5}{4}} - \frac{1}{2}$, Corollary 6.42 implies $k_2 < \sqrt{h+\frac{5}{4}} - \frac{1}{2}$. Since T is optimal, Lemma 6.40 yields $k_i(1+k_1) \leq k_2(1+k_1) \leq h+1$. Thus

$$k_i \leq \frac{h+1}{k_1+1}.$$

Because $k_1 > \left\lceil \sqrt{h+\frac{5}{4}} - \frac{1}{2} \right\rceil$, then necessarily $k_2 < k_1 - 1$. Corollary 6.41 gives the following:

$$k_1(1+k_i) \geq k_1(1+k_k) \geq h+1.$$

This is equivalent to

$$k_i \geq \frac{h+1}{k_1} - 1.$$

■

Remark 6.46. *Let T be an optimal tree with n vertices and height h such that each T_i is a k_i-split with $k_1 \geq k_2 \geq \ldots \geq k_r$ which falls into one of the following three categories:*

1. *(Paths) $k_r = 0$, $k_{r-1} \in \{0,1\}$, $k_{r-2} = \ldots = k_1 = 1$.*
2. *(One large) $k_1 > \left\lceil \sqrt{h+\frac{5}{4}} - \frac{1}{2} \right\rceil$ and $k_i = \left\lfloor \frac{h+1}{k_1+1} \right\rfloor$ for each $i \in \{2,\ldots,r\}$ provided $\left\lfloor \frac{h+1}{k_1+1} \right\rfloor \geq \frac{h+1}{k_1} - 1$.*

3. (Even distribution) $k_1 \leq \left\lceil \sqrt{h + \frac{5}{4}} - \frac{1}{2} \right\rceil$ *and for all* $i, j \in [r]$, $|k_i - k_j| \leq 1$.

Provided that the k_i-values in an optimal tree follow an even distribution, we are interested in determining the optimal value for $deg(\rho) = r$.

Lemma 6.47. *For fixed* $h, n \in \mathbb{Z}$, *let* T *be an optimal tree with* n *vertices, height* h, *and where each* T_i *is a* k_i*-split. Fix* $t \in \mathbb{R}$, $t \geq 2$ *which satisfies the inequality* $h^{1/(t+1)} > \ln(6h)$. *For* $x \in \left[h^{1/(t+1)}, h^{1/t}\right]$ *with* $n \geq (h + x)(h + x - 1) + 1$, *then*

$$|\{i : k_i = x\}| < h + x - 1.$$

Proof. Let T be a tree with root degree r and each T_i is a k_i-split. Suppose for contradiction that $k_1 = k_2 = \ldots = k_{h+x-1} = x$ (where the k_i values are not necessarily in non-increasing order).

Let H be the subtree induced by $T_1, \ldots, T_{h+x-1}$ and the root ρ. Here, each T_i is an x-split. Thus the number of root-containing subtrees in H is

$$F_H(\rho) = (h + x^2 + x + 1)^{h+x-1}.$$

Let T_i' be an $(x-1)$-split. Define a new tree T' by replacing T_i with T_i' for each $i \in [h + x - 1]$ and increasing the degree of the root by one so that the new branch T_0' is also an $(x-1)$-split. Let H' be the subtree induced by $T_0', T_1', \ldots, T_{h+x-1}'$ and the root of T'. The number of root-containing subtrees in H' is

$$F_{H'}(\rho) = (h + x^2 - x + 1)^{h+x}.$$

In order to compare the number of root-containing subtrees of T and T', it suffices to compare the number of root-containing subtrees of H and H'.

In order to compare these, consider the ratio:

$$\begin{aligned}
\frac{F_T(\rho)}{F_{T'}(\rho)} =& \frac{F_H(\rho)}{F_{H'}(\rho)} \\
=& \frac{(h+x^2+x+1)^{h+x-1}}{(h+x^2-x+1)^{h+x}} \\
=& \frac{1}{h+x^2-x+1}\left(1+\frac{2x}{h+x^2-x+1}\right)^{h+x} \\
\geq& \frac{1}{h+h^{2/t}+h^{1/t}+1}\left(1+\frac{2h^{1/(t+1)}}{h+h^{2/t}-h^{1/(t+1)}+1}\right)^{h+h^{1/(t+1)}} \\
& \qquad \text{since } x \in [h^{1/(t+1)}, h^{1/t}] \\
=& \frac{1}{h+h^{2/t}+h^{1/t}+1}\left(1+\frac{2}{h^{t/(t+1)}+h^{(t+2)/(t^2+t)}-1+h^{-1/(t+1)}}\right)^{h+h^{1/(t+1)}} \\
\geq& \frac{1}{3h}\left(1+\frac{2}{h^{t/(t+1)}+h^{(t+2)/(t^2+t)}}\right)^{h} \qquad \text{for } t \geq 2 \\
\geq& \frac{1}{3h}\left(1+\frac{2}{h^{t/(t+1)}+h^{t/(t+1)}}\right)^{h^{t/(t+1)}h^{1/(t+1)}} \\
=& \frac{1}{3h}\left(1+\frac{1}{h^{t/(t+1)}}\right)^{h^{t/(t+1)}h^{1/(t+1)}} \\
\geq& \frac{1}{3h}\cdot\frac{1}{2}e^{h^{1/(t+1)}} \\
>& 1. \qquad \text{since } h^{1/(t+1)} > \ln(6h)
\end{aligned}$$

As a result, T is not an optimal tree because T' also has n vertices and height h but has fewer subtrees which contain its root. ∎

We obtain the following corollary:

Corollary 6.48. *Fix $h \geq 550$ and $n \in \mathbb{Z}$ with $n \geq 6h^2$. Let T be an optimal tree with subtrees T_i which are k_i-splits. For any $x \in \left(\ln(6h), \sqrt{h}\right]$, then*

$$|\{i : k_i = x\}| < h + x - 1.$$

Proof. Fix n, h with $n \geq 6h^2$. Since $h \geq 550$, $\ln(6h) < h^{1/3}$. Suppose a tree T has root degree r. Let $x \in \left(\ln(6h), \sqrt{h}\right]$. If $\ln(6h) < x \leq h^{1/3}$, there is $t \in \mathbb{R}$ such that

$x = h^{1/(t+1)}$ ($t = \frac{\ln h}{\ln x} - 1$) and $t \geq 2$. If $h^{1/3} < x \leq h^{1/2}$, let $t = 2$ in which case $h^{1/(t+1)} = h^{1/3} > \ln(6h)$ for $h \geq 550$. ∎

In the statement of Corollary 6.48, we require $n \geq 6h^2$. First this ensures that T can possibly have $(h + x - 1)$ subtrees which are x-splits for $x \leq \sqrt{h}$. On the other hand, Corollary 6.43 says $|k_i - k_j| \leq 1$. If $n \geq 6h^2$, then we guarantee that T will have at least $h + x - 1$ subtrees T_i, each of which is an x-split, even when $x = \sqrt{h}$.

Remark 6.49. *Therefore if $n \geq 6h^2$ with $h \geq 550$, then each $k_i \leq \ln(6h) + 1$. Even in the "one large" case, the value of k_1 must be large enough so that $k_i = \left\lfloor \frac{h+1}{k_1+1} \right\rfloor \leq \ln(6h) + 1$.*

While we do not yet have a complete characterization of the optimal trees, we have established many of their structural properties to guide our continued study on this topic.

Chapter 7

Some Remarks on Baranyai's Theorem

About 40 years ago, Baranyai (1973) found a proof using network flows for a long-standing open problem about set partitions: if $k|n$, then all $\binom{n}{k}$ k-element subsets can be partitioned into $\binom{n-1}{k-1}$ families, such that each family is a partition of the n-element underlying set. The proof uses a polynomial time algorithm to develop such a partition. However, the algorithm sheds little light on the construction. Before Baranyai's proof, the existence for $k = 2$ was well-known and easily constructible as seen in Fig. 7.7 taken from Lint and Wilson (1996). Peltesohn (1936) proved the existence for $k = 3$ by combinatorial arguments. For $k = 3$ an algebraic construction also exists due to Beth (1974). While no explicit construction has been found for larger sets, such a construction may prove enlightening about finite sets as Baranyai's theorem is a strong result. For example, it implies the Erdős-Ko-Rado theorem in one line for $k|n$.

Erdős and Székely (1989) was the first to establish a bijection between set partitions and rooted leaf-labelled trees. It is our hope that working with trees instead of partitions will provide the insight needed to develop a more general construction of Baranyai partitions. We use here another bijection found by Stanley (1999) Ex. 5.43 to provide an alternative construction for the $k = 2$ case of Baranyai's Theorem. Eventually one could describe our constructions without trees, but we think that trees give a strong motivation for them. We also prove that our construction differs from the standard construction shown on Fig. 7.7.

7.1 Baranyai's Theorem

In order to state Baranyai's Theorem, we first need the following definitions.

Definition 7.1. *Given the set* $[n] := \{1, 2, \ldots, n\}$*, and* $k \in [n]$

1. $\binom{[n]}{k}$ *is the collection of subsets of* $[n]$ *having size* k*.*
2. $S \subset \binom{[n]}{k}$ *is a* k*-partition of* $[n]$ *provided the elements of* S *form a partition of* $[n]$*.*

Theorem 7.2 (Baranyai, 1975)**.** *If* k *divides* n*, then there is a partition of* $\binom{[n]}{k}$ *into* $r = \frac{k}{n}\binom{n}{k}$ *rows, each of which is a* k*-partitions. We call this an* (n, k)*-Baranyai partition.*

Baranyai proved the existence of these (n, k)-Baranyai partitions for integers k and n with $k|n$. We now turn our attention to the case when $k = 2$ and work toward a constructive proof.

7.2 Bijection with binary trees

Definition 7.3. *A* binary rooted tree *is a vertex rooted tree where each vertex has 0 or 2 children, including the root. One that is* labeled *bears distinct labels for the leaves. If it is* unordered*, there is no ordering given to the two children of a single vertex.*

Let $\mathcal{B}_n$ *be the collection of labeled, unordered, binary rooted trees with* n *non-root vertices. Each will have* ℓ_n *leaves and* $\ell_n - 2$ *non-root, non-leaf vertices. Thus* $n = \ell_n + (\ell_n - 2)$*. The leaves will have label set* $\{1, \ldots, \ell_n\}$*.*

Stanley defines a bijection between $\mathcal{B}_n$ and the 2-partitions of $[n]$. The bijection is defined as follows: Start with $T \in \mathcal{B}_n$. Extend the labeling to all of the non-root vertices by iterations of the following procedure.

Algorithm 7.4. *Among the vertices which are unlabeled, non-root vertices with children which are both labeled, chose the one whose child has the least label of those considered. If labels* $1, \ldots, s$ *have already been used, apply the label* $s+1$ *to the selected vertex.*

This will result in a labeling of the whole tree with labels $1, 2, \ldots, n$ as exemplified in Fig. 7.1. Since we have a binary tree, we can create a 2-partition of $[n]$ by pairing the labels of vertices which have the same parent. Because the tree is unordered, the 2-partition will be unique.

Likewise, each 2-partition can be described by a unique tree in $\mathcal{B}_n$. Induce an ordering on the pairs based on the maximum element in each. Start with a root vertex and two leaves. These leaves will obtain the labels from the first (largest) pair, i.e. the one containing n. Below the leaf with label $n-i$, hang two new leaves with labels from the $i+2$ pair in the ordering, starting with $i=0$. Ignoring the non-leaf labels in the final product, we have a tree in $\mathcal{B}_n$. Mutatis mutandis, the bijection extends for non-binary trees.

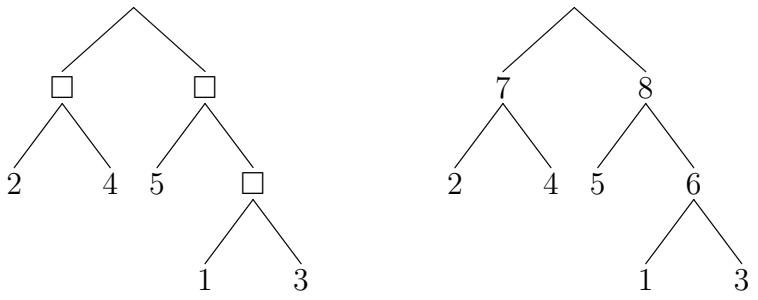


Figure 7.1 A leaf-labeled binary tree which corresponds to partition 13|24|56|78.

7.3 Tree construction

For even n, the goal is to describe a collection of $\frac{2}{n}\binom{n}{2} = n-1$ binary trees which correspond to an $(n,2)$-Baranyai partition. The construction is inductive on n. As a

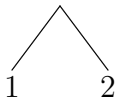


Figure 7.2 The single binary tree for the $(2,2)$-Baranyai partition.

base case, when $n = 2$ there is only one binary tree (in Fig. 7.2) with leaves labeled $\{1, 2\}$ corresponding to the unique $(2, 2)$-Baranyai partition.

Assume for all even $m < n$ there is a collection of $m - 1$ trees from $\mathcal{B}_m$ which corresponds to an $(m, 2)$-Baranyai partition by the bijection described above. Recall that each tree in the collection will have m non-root vertices and ℓ_m leaves labeled $1, \ldots, \ell_m$. Extending the labeling, the interior vertices take labels $\ell_m + 1, \ldots, m$. For ease of notation, let $\mathcal{T}_m$ name this collection of $m - 1$ fully label trees.

In the inductive step, we create $\mathcal{T}_n$, a collection of $n - 1$ trees from $\mathcal{B}_n$ whose corresponding partitions create an $(n, 2)-$Baranyai partition. The construction is broken into two cases

$$\text{(Case 1)} \quad n \equiv 0 \mod 4$$

$$\text{(Case 2)} \quad n \equiv 2 \mod 4$$

Case 1: $n \equiv 0 \mod 4$ (i.e. $n = 2j$ where j is even)

We require $n - 1$ binary trees with $\ell_n = j + 1$ leaves and $j - 1$ non-root internal vertices. The method to extend labelings guarantees that each tree will correspond to a 2-partition of $[n]$. Because n is finite, it suffices to check that every element of $\binom{[n]}{2}$ appears in some partition. The result will be an $(n, 2)-$Baranyai partition.

The collection $\mathcal{T}_n$ will be the union of two sets. The first collection, $\mathcal{T}_n^1$, will consist of $j - 1$ trees which account for all the 2-subsets of $[n]$ in which both elements come from $\{1, \ldots, j\}$ or both elements come from $\{j + 1, \ldots, n\}$. The second collection $\mathcal{T}_n^2$, of j trees will exhibit all the 2-subsets with one element from $\{1, \ldots, j\}$ and the

other from $\{j+1,\ldots,n\}$. Thus $\mathcal{T}_n^1 \cup \mathcal{T}_n^2$ will be the desired collection of $n-1$ trees in which all 2-subsets of $[n]$ appear in exactly one tree.

Creating $\mathcal{T}_n^1$*:* Start with $\mathcal{T}_j$, the collection of $j-1$ trees from the induction hypothesis. The non-root vertex set takes labels $1,\ldots,j$ while the leaves are labeled $1,\ldots,\ell_j$. Increase each of the labels by j. The vertex labels are now $j+1,\ldots,n$ and the leaves are labeled $j+1,\ldots,j+\ell_j$. Hang cherries (pairs of leaves) from the existing leaves with labels $\{j+2,\ldots,j+\ell_j\}$. This creates $2(\ell_j-1)=j$ new leaves which need labels. Let $\mathcal{T}_j'$ name this collection of partially labeled binary trees. Fig. 7.3 demonstrates the process on a tree from $\mathcal{T}_4$.

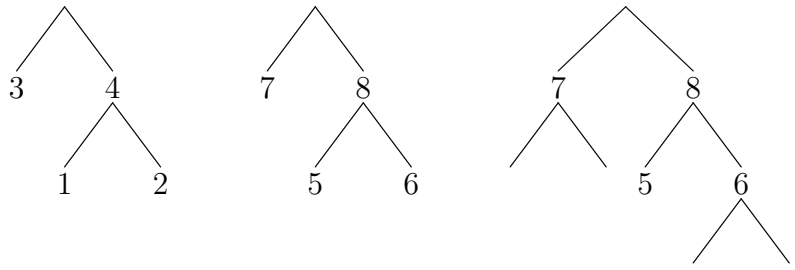


Figure 7.3 The left tree is from $\mathcal{T}_4$. Adding 4 to each of the labels yields the middle tree. Finally we hang cherries from 3 of the leaves to obtain the right-most tree in $\mathcal{T}_4'$.

Each $T' \in \mathcal{T}_j'$ was constructed from a tree $T \in \mathcal{T}_j$. For the 2-partition that corresponds to T, induce a well-ordering on the pairs according to their least element. The pair containing 1 will be the first (least) pair. We then use these pairs to label the unlabeled leaves of T'. Specifically, the ith pair will label the children of the vertex labeled $j+i+1$. Labeling in this order respects the algorithm for extending labelings.

By construction, this collection of $j-1$ trees accounts for all of the $\binom{j}{2}$ 2-subsets of $\{1,\ldots,j\}$ and all of the $\binom{j}{2}$ 2-subsets from $\{j+1,\ldots,n\}$ with $\frac{j}{2}$ of each type in each tree.

Creating $\mathcal{T}_n^2$: We utilize j caterpillar binary trees (as seen in Fig. 7.4) to account for the pairs in which one element is from $\{1, \ldots, j\}$ and the other is from $\{j+1, \ldots, n\}$. On every tree, apply label $j+1$ to one of the leaves farthest from the root. Necessarily, the internal vertices are labeled $j+2, \ldots, n$ increasing toward the root.

For the complete bipartite graph with vertex classes $\{1, \ldots, j\}$ and $\{j+1, \ldots, n\}$, we can partition the edges into j perfect matchings since the graph is regular. Using one matching for each caterpillar tree, label the vertices so that pairs of vertices with the same parent are exactly the pairs in the matching. This completes the construction of $\mathcal{T}_n^2$.

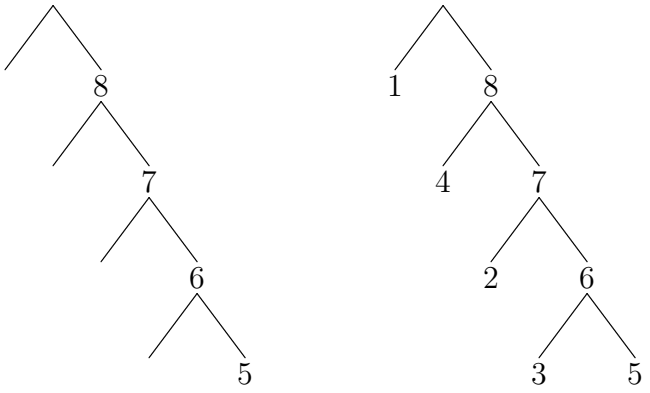


Figure 7.4 The left tree is the general caterpillar tree. The right tree corresponds to the matching 35|26|47|18.

Between $\mathcal{T}_n^1$ and $\mathcal{T}_n^2$, we have constructed a total of $j - 1 + j = n - 1$ trees with each pair from $\binom{[n]}{2}$ appearing in exactly one tree. This completes our construction of $\mathcal{T}_n$ when $n \equiv 0 \mod 4$.

Case 2: $n \equiv 2 \mod 4$ (i.e. $n = 2j$ where $j > 1$ is odd)

We need $n - 1$ trees from $\mathcal{B}_n$, each with $\ell_n = j + 1$ leaves and $j - 1$ non-leaf, non-root vertices. As in Case 1, this is done in two steps. First, we create $\mathcal{T}_n^1$ with $j - 2$ trees to account for all pairs from $\{j + 2, \ldots, n\}$ and all but $j + 1$ pairs from $\{1, \ldots, j + 1\}$. The missing $j + 1$ pairs make two 2-partitions P_a and P_b of

$\{1, \ldots, j+1\}$. Second, we create $\mathcal{T}_n^2$ to account for the pairs from P_a and P_b in addition to the $(j+1)(j-1)$ pairs in which one element is from $\{1, \ldots, j+1\}$ and the other element is from $\{j+2, \ldots, n\}$.

Creating $\mathcal{T}_n^1$*:* Because $j-1$ is even, the induction hypothesis gives the collection $\mathcal{T}_{j-1}$ of $j-2$ trees, each with ℓ_{j-1} leaves. We now modify these trees to create trees with n non-root vertices.

On each tree of $\mathcal{T}_{j-1}$, increase all of the labels by $j+1$ so that the new labels are $j+2, \ldots, n$. At the end of each leaf, hang a pair of leaves. This yields exactly $2\ell_{j-1} = 2(\frac{j-1}{2}+1) = j+1$ new leaves for a total of n non-root vertices. Call this collection of $j-2$ trees $\mathcal{T}'_{j-1}$. Figure 7.5 represents the procedure for $n = 10$.

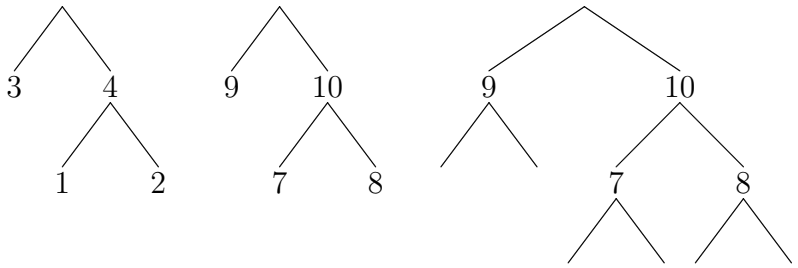


Figure 7.5 The left tree is from $\mathcal{T}_4$. The middle tree results from increasing the label values by 6. The right tree is the outcome in $\mathcal{T}'_6$ after hanging 3 pairs of new leaves.

It remains to label the $j+1$ new leaves. Because $j+1$ is even, the induction hypothesis supplies $\mathcal{T}_{j+1}$ corresponding to an $(n,2)-$Baranyai partition. Let T_a and T_b be two arbitrarily chosen trees in $\mathcal{T}_{j+1}$ with corresponding partitions P_a and P_b. Fix any bijection from $\mathcal{T}'_{j-1}$ to $\mathcal{T}_{j+1} \setminus \{T_a, T_b\}$. Consider an arbitrary pair (T', T) in the bijection, $T' \in \mathcal{T}'_{j-1}$ and $T \in \mathcal{T}_{j+1}$. Induce a well-order on the pairs associated with T according to the least element in each pair. Now in T', label the children of the vertex labeled $j+1+i$ with the values in the ith pair where the pair containing 1 is the first pair. This algorithm results in a labeling of all the leaves of T' that respects the method for extending labelings.

The construction of $\mathcal{T}_n^1$ is complete with $j-2$ trees containing $\binom{j-1}{2}$ pairs from $\{j+2,\ldots,n\}$ and $(j-2)\frac{j+1}{2}$ pairs from $\{1,\ldots,j+1\}$. The remaining $j+1$ pairs from $\{1,\ldots,j+1\}$ are those in P_a and P_b. They will be accounted for in $\mathcal{T}_n^2$.

Creating $\mathcal{T}_n^2$: The remaining pairs will be realized in $j+1$ caterpillar binary trees. The internal vertices will necessarily be labeled $j+2,\ldots,n$, increasing as we move closer to the root as in Fig. 7.6.

We can use matching theory to label the leaves of the caterpillar trees. First, create a complete bipartite graph with vertices $\{1,\ldots,j+1\}$ on the left and vertices $\{j+2,\ldots,n,x,y\}$ on the right.

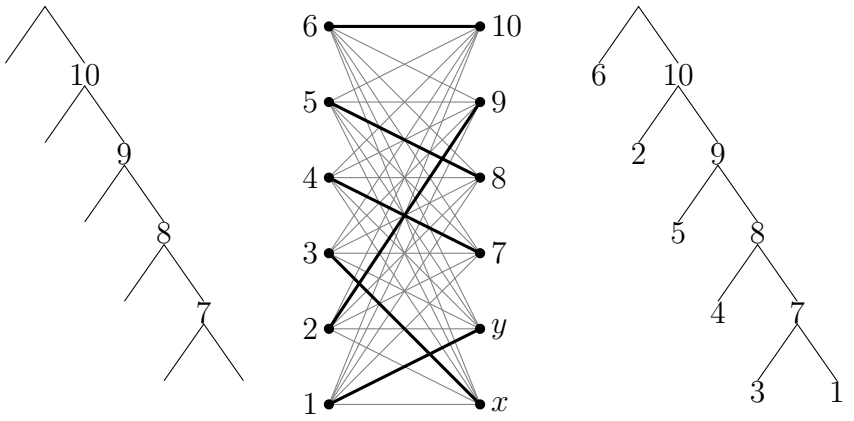


Figure 7.6 The left caterpillar is the general layout for $n=10$ with leaf labels needed. The middle bipartite graph shows a perfect matching which extends the matching $\{3,x\},\{1,y\}$ for $\{1,3\}$ in P. The right tree is the caterpillar corresponding to the matching.

For $P=P_a\cup P_b$, start with a single pair $\{k,\ell\}\in P$. Match k with x and ℓ with y. The remaining graph is regular bipartite, so we can extend our matching to a perfect matching M which will determine the leaf labels of a single caterpillar. For all $i\in\{j+2,\ldots,n\}$, the vertex matched with i will have the same parent as i in the caterpillar. The vertices matched with x and y will label the two leaves farthest from the root.

Removing the edges of M from the bipartite graph, regularity is maintained. So we repeat the procedure to obtain the next caterpillar tree. You may notice that each value of $\{1, \ldots, j+1\}$ appears in exactly two pairs of P. So for $\{\ell, m\} \in P$, ℓ will be paired with x since it was previously paired with y. Thus there will be no problem with representing all pairs of P. Once the bipartite graph has been decomposed into $j+1$ matchings, we will have our $j+1$ caterpillar trees, completing our construction of $\mathcal{T}_n^2$.

With $\mathcal{T}_n^1$ and $\mathcal{T}_n^2$, we have constructed $(j-2)+(j+1) = 2j-1 = n-1$ trees for the $(n,2)$-Baranyai partition.

7.4 New $(n,2)-$Baranyai partitions

The easy known algorithm to create $(n,2)$-Baranyai partitions is as follows:

Algorithm 7.5. *Choose one element of $\{1,2,\ldots,n\}$ to be the center. Display the remaining $n-1$ values equidistant in any order on a circle around the center. The procurement of the partitions is best described first with an example.*

In Fig. 7.7, we have 8 in the center with $\{1,2,\ldots,7\}$ in increasing order around the circle. Connecting 8 with 1, the remaining integers are paired as in the left diagram to create 2-partition $\{\{1,8\},\{2,7\},\{3,6\},\{4,5\}\}$. By connecting 8 with 2 and matching the remaining integers in a similar way, we obtain the 2-partition $\{\{2,8\},\{1,3\},\{4,7\},\{5,6\}\}$. Continue to rotate the pairing configuration around the circle to obtain 7 different 2-partitions which contain each element of $\binom{[8]}{2}$ exactly once. This is an $(8,2)-$Baranyai partition.

Algorithm 7.6. *In general, to obtain a partition into 2-sets, match the center with an arbitrary entry on the circle, and after that match those pairs whose connecting line is perpendicular to the radius connecting the center to the selected entry.*

Figure 7.7 A circular representation to find an $(8, 2)$-Baranyai partitions (Lint and Wilson 1996).

Claim 7.7. *The construction developed in this paper does not arise from the circular construction described here for any* $4 \leq n \neq 10$*, for any relabeling the elements on Fig. 7.7.*

Proof. We will break this proof into two cases according to the two cases for the tree construction.

Case 1: When $n \equiv 0 \mod 4$, $n = 2j$, represent the elements from $\{1, \ldots, j\}$ with color blue, b, and the elements from $\{j+1, \ldots, n\}$ with color red, r. Note that blue and red occur the same number of times. Recall each 2-partition in $\mathcal{T}_n$ from the tree construction is one of the following types:

Type I: All pairs have one element from the set $\{1, \ldots, j\}$ and one element from $\{j+1, \ldots, n\}$, i.e. all pairs are of the form rb.

Type II: Half of the pairs have both elements from $\{1, \ldots, j\}$ and the other half of the pairs have both elements from $\{j+1, \ldots, n\}$, i.e. half of the pairs are bb and the other half are rr.

For contradiction, assume there is a circular representation of $\{1, \ldots, n\}$ whose corresponding 2-partitions exactly match those of $\mathcal{T}_n$.

At this point, we assume the integer in the center is red. (Identical arguments work if the integer in the center is blue.) It is also safe to assume that there are two

consecutive positions on the circle, one red and the other blue (here we use $n > 2$). The following deductions can be seen Fig. 7.8.

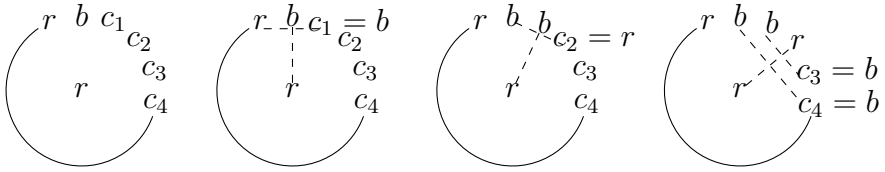


Figure 7.8 For Case 1, this shows the steps to determine the colors on circle.

The 2-partition that arises when the center r is paired with the b must be of Type I. In this same partition, r will be paired with c_1, forcing it to be blue because this is Type I.

Now let's consider the next 2-partition obtained by pairing the middle r with blue c_1. This again puts us into Type I. Based on the circular arrangement, c_2 will be paired with b and thus must be red.

Looking at one more 2-partition, where the middle r is paired with red c_2, we find ourselves in Type II. Because blue c_1 will be paired with c_3, it must be blue. Similarly, c_4 is blue.

The pattern $rbbrbbr\ldots$ repeats around the circle. However, half of the elements should be red, posing a contradiction.

Case 2: When $n \equiv 2 \mod 4$, $n = 2j$, color the elements of $\{1, \ldots, j+1\}$ blue, b, and the elements of $\{j+2, \ldots, n\}$ red, r. This time, there are two more blue than red. Each 2-partition of $[n]$ in the tree construction is one of the following types:

Type I: There is one pair with both elements in $\{1, \ldots, j+1\}$ and the rest have one element from $\{1, \ldots, j+1\}$ and one element from $\{j+2, \ldots, n\}$, i.e. one is bb and all others are rb.

Type II: All pairs have either have both elements from $\{1, \ldots, j+1\}$ or both elements from $\{j+2, \ldots, n\}$, i.e. all pairs are rr or bb.

Because there are 2 more blue integers than red, we must find the sequence rbb somewhere on the circle. This time we split the problem into two cases based on the color of the middle integer.

Case 2/Red: Suppose the middle integer is red. When the middle r is paired with the r on the circle, as in Fig. 7.9, we have a Type II partition. Thus d_1 and d_2 are blue. Now c_1 could be red or blue. So we consider the two possibilities separately.

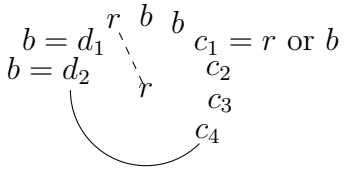


Figure 7.9 In Case 2/Red, we start with a red middle. By pairing r with r, we obtain more color information.

If c_1 is blue, then pairing the middle r with the b as in Fig. 7.10, we find ourselves in Type I. There is already a bb pair, so c_2 is blue and c_3, c_4 are red. However, when we pair the middle r with the next b we find an rr in the same 2-partition which is not possible in either Type I or Type II. Thus we have a contradiction.

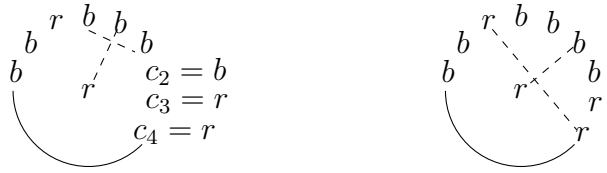


Figure 7.10 When c_1 is blue, we obtain a contradiction.

If c_1 is red, then pairing the middle r with c_1 puts us in Type II. Thus c_2 and c_3 are blue while c_4 is red as in Fig. 7.11. Next, pairing the middle r with c_4 puts us in Type II again and continues the $rbbrbb$ pattern around the circle. Repeating this procedure, we see that there are twice as many blues as reds on the circle which contradicts the size of the red and blue sets for $n \neq 10$.

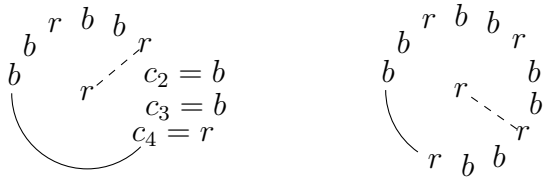


Figure 7.11 When c_1 is red, these are the deductions made.

Case 2/Blue: When the middle integer is blue, the color of c_1 is not yet determined. So we consider each color possibility separately.

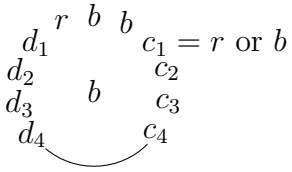


Figure 7.12 The set-up for Case 2/Blue.

If c_1 is red, pair the center b with the two blues on the circle, one at a time as in Fig. 7.13. Both are Type I with the pair containing the center as the exceptional bb pair in each. All other pairs in these will be rb. The first forces d_1 to be blue while the second makes c_2 blue. Using this new information, return to the first pairing to see d_2 is red and the second pairing to see c_3 is red. In this way, we find that the colors alternate around the rest of the circle.

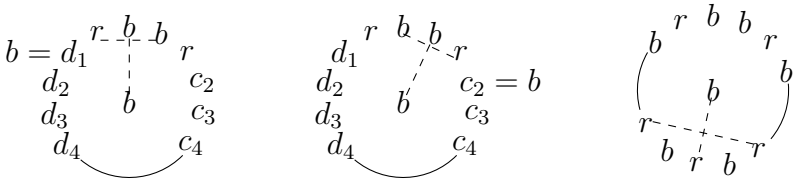


Figure 7.13 When c_1 is red, the first two pairings work alternately to obtain the configuration on the right.

Connecting the center b to the red on the opposite side of the circle, as in Fig. 7.13, we find a single partition with both rb and rr which poses a contradiction.

If, on the other hand, c_1 is blue. Pairing the center b with the middle blue on the circle (left picture of Fig. 7.14) puts us in Type II because there will be two bb pairs. Therefore c_2 is red. Next, pairing the center b with the rightmost blue puts us in Type I because there is an rb pair. Since there is already a bb pair, all the rest must be rb. Therefore c_3 will be red and c_4 will be blue.

Back to the first pairing, we can conclude d_1 is red. And the second pairing gives c_5 is blue. However, as in the last picture of Fig. 7.14, there is now a pairing with an rb and two bbs which is neither Type I nor Type II posing another contradiction.

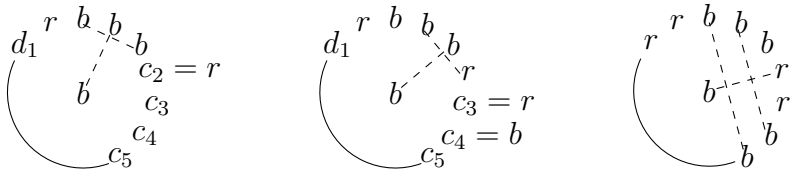


Figure 7.14 When c_1 is blue, these conclusions can be made.

This completes the proof that the 2-partitions constructed using the tree algorithm are distinct from those obtained by the circular arrangement algorithm. ∎

7.5 Conclusion

There are many different bijections between trees and partitions. As we continue on this project of constructing Baranyai partitions, we explore other bijections in a search for constructions which could be extended for larger values of k.

Bibliography

Andriantiana, E., S. Wagner, and H. Wang (2013). "Greedy trees, subtrees and antichains." In: *Electron. J. Combin.* 20 (3), pp. 1–25.

Bach, E. and J. Shallit (1996). *Algorithmic Number Theory, Volume I: Efficient Algorithms.* MIT Press.

Baranyai, Zs. (1973). "On the factorization of the complete uniform hypergraph." In: *Infinite and Finite Sets*. Vol. 1. Colloquia Mathematica Societatis Janos Bolyai. Amsterdam, Netherlands: North Holland Publishing Company, pp. 91 –107.

Barefoot, C.A., R.C. Entringer, and L.A. Székely (1997). "Extremal values for ratios of distances in trees." In: *Discrete Appl. Math.* 80 (1), pp. 37–56.

Bartlett, M., E. Krop, C. Magnant, F. Mutiso, and H. Wang (2014). "Variations Of Distance-Based Invariants Of Trees." In: *Combin. Math. Combin. Comput.* 91, pp. 19–29.

Beth, T. (1974). "Algebraische Auflösungsalgorithmen für einige unendliche Familien von 3-Designs." In: *Matematiche* 29, pp. 105–135.

Brightwell, G. and P. Winkler (1991). "Counting linear extensions." In: *Order* 8, pp. 225–242.

Çela, E., N. Schmuck, S. Wimer, and G.J. Woeginger (2011). "The Wiener maximum quadratic assignment problem." In: *Discrete Optim.* 8, pp. 411–416.

Cook, S. A. (1971). "The Complexity of Theorem-proving Procedures." In: *Proceedings of the Third Annual ACM Symposium on Theory of Computing*. STOC '71. New York, NY, USA: ACM, pp. 151–158.

Dankelmann, P., W. Goddard, and C. Swart (2004). "The average eccentricity of a graph and its subgraphs." In: *Util. Math.* 65, pp. 41–51.

Dankelmann, P. and S. Mukwembi (2014). "Upper bounds on the average eccentricity." In: *Discrete Appl. Math.* 167, pp. 72–79.

Dobzhansky, T. and A. H. Sturtvant (1938). "Inversions in the Chromosomes of Drosophila Pseudoobscura." In: *Genetics* 23.1, pp. 28–64.

Entringer, R., D. Jackson, and D. Snyder (1976). "Distance in Graphs." In: *Czechoslovak Math. J.* 26 (2), pp. 283–296.

Erdős, P. L. and L. A. Székely (1989). "Applications of antilexicographic order I: An enumerative theory of tree." In: *Adv. in Appl. Math.* 10, pp. 488 –496.

— (1994). "On weighted multiways cuts in trees." In: *Math. Program.* 65 (1-3), pp. 93 –105.

Feijão, P. and J. Meidanis (2011). "SCJ: A Breakpoint-Like Distance that Simplifies Several Rearrangement Problems." In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 8.5, pp. 1318–1329.

Fitch, W. M. (1971). "Toward Defining the Course of Evolution: Minimum Change for a Specific Tree Topology." In: *Systematic Zoology* 20.4, pp. 406–416. ISSN: 00397989.

Gill, J. (1977). "Computational complexity of probabilistic Turing machines." In: *SIAM Journal of Computing* 6.4, pp. 675–695.

Jordan, C. (1869). "Sur les assemblages de lignes." In: *J. Reine Angew. Math.* 70, pp. 185–190.

Karzanov, A. and L. Khachiyan (1991). "On the conductance of order Markov chains." In: *Order* 8.1, pp. 7–15.

Lint, J. H. van and R. M. Wilson (1996). *A Course in Combinatorics.* Cambridge University Press.

Lovász, L. (2007). *Combinatorial Problems and Exercises.* 2nd ed. Providence, Rhode Island: AMS Chelsea Publishing.

Mélykúti, B. (2006). "The Mixing Rate of Markov Chain Monte Carlo Methods and some Applications of MCMC Simulation in Bioinformatics." MA thesis. Eötvős Loránd University.

Metropolis, N., A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller (1953). "Equations of state calculations by fast computing machines." In: *Journal of Chemical Physics* 21.6, pp. 1087–1092.

Miklós, I., Sándor Z. Kiss, and E. Tannier (2014). "Counting and sampling SCJ small parsimony solutions." In: *Theoret. Comput. Sci.* 552, pp. 83 –98.

Palmer, J.D. and L.A. Herbon (1988). "Plant mitochondrial DNA evolves rapidly in structure, but slowly in sequence." In: *Journal of Molecular Evolution* 28, pp. 87–97.

Papadimitriou, C. (1994). *Computational Complexity.* Addison-Wesley.

Peltesohn, R. (1936). "Das Turnierproblem für Spiele zu je dreien." PhD thesis.

Rosser, B. (1941). "Explicit bounds for some functions of prime numbers." In: *Amer. J. Math.* 63.1, pp. 211–232.

Sankoff, D. and P. Rousseau (1975). "Locating the vertices of a Steiner tree in an arbitrary metric space." In: *Math. Prog.* 9.1, pp. 240–246. ISSN: 0025-5610. DOI: 10.1007/BF01681346.

Schmuck, N., S Wagner, and H. Wang (2012). "Greedy trees, caterpillars, and Wiener-type graph invariants." In: *MATCH Commun. Math. Comput. Chem.* 68, pp. 273–292.

Sills, A. and H. Wang (2015). "The minimal number of subtrees of a tree." In: *Graphs Combin.* 31 (1), pp. 255–264.

Stanley, R. (1999). *Enumerative Combinatorics.* Vol. 2. Cambridge University Press.

Sturtvant, A. H. (1913). "The linear arrangement of six sex-linked factors in Drosophila, as shown by their mode of association." In: *Journal of Experimental Zoology* 14, pp. 43–59.

— (1921). *The North American species of Drosophila.* The Carnegie Institution of Washington.

Székely, L.A. and H. Wang (2005). "On subtrees of trees." In: *Adv. in Appl. Math.* 34, pp. 138 –155.

— (2007). "Binary trees with the largest number of subtrees." In: *Discrete Appl. Math.* 155, pp. 374–385.

— (2013). "Extremal values of ratios: distance problems vs. subtree problems in trees." In: *Electron. J. Combin.* 20 (1), pp. 1–20.

— (2014). "Extremal values of ratios: distance problems vs. subtree problems in trees II." In: *Discrete Math.* 322, pp. 36–47.

Valiant, L.G. (1979). "The complexity of computing the permanent." In: *Theo. Comp. Science* 8.2, pp. 189 –201.

Wang, H. (2008). "The extremal values of the Wiener index of a tree with given degree sequence." In: *Discrete Appl. Math.* 156, pp. 2647–2654.

— (2014). "The distances between internal vertices and leaves of a tree." In: *European J. Combin.* 41, pp. 79–99.

Wei, W. D. (1982). "The class $\mathcal{U}(R, S)$ of (0,1) matrices." In: *Discrete Math.* 39, pp. 201–205.

Welsh, D. (1993). *Complexity: Knots, Colourings and Countings.* London Mathematical Society Lecture Note Series 186. Cambridge University Press.

Zhang, X.D., Q.Y. Xiang, L.Q. Xu, and R.Y. Pan (2008). "The Wiener index of trees with given degree sequences." In: *MATCH Commun. Math. Comput. Chem.* 60, pp. 623–644.

Zhang, X.M. and X.D. Zhang (2015). "Minimal number of subtrees with a given degree sequence." In: *Graphs Combin.* 31 (1), pp. 309–318.

Zhang, X.M., X.D. Zhang, D. Gray, and H. Wang (2013). "The number of subtrees of trees with given degree sequence." In: *J. Graph Theory* 73, pp. 280–295.

Appendix A

Method for Selecting the Binary Strings in $\mathcal{C}_i$

For a D3CNF Γ, Subsection 2.1.1 defines a set $\mathcal{C}_i$ of 50 binary strings that had number of useful properties needed to encode clauses of Γ. In this section, we explain the method by which we selected these 50 strings.

Fix Γ with k clauses and n literals. Fix a clause $c_i = v_\alpha \vee v_\beta \vee v_\gamma$ in Γ with $\alpha \neq \beta \neq \gamma \neq \alpha$. The goal is to define a multiset of binary strings $\mathcal{C}_i = \{\nu_j^i\}_{j=1}^m$ taken from the set

$$S = \{0,1\}^{2n+t}$$

with coordinates

$$(x_1, y_1, x_2, y_2, \ldots, x_n, y_n, e_1, e_2, \ldots, e_t)$$

so that the multiset has the following properties:

1. $\mathcal{M}(\nu_1^i, , \ldots, \nu_m^i) = \{0,1\}^{2n} \times \{0\}^t$
2. For any $\mu, \mu' \in \mathcal{M}'_{c_i}$ and $\eta, \eta' \in \mathcal{M}' \setminus \mathcal{M}'_{c_i}$, then

$$\{H(\mu, \nu_j^i) : j \in [m]\} = \{H(\mu', \nu_j^i) : j \in [m]\},$$
$$\{H(\eta, \nu_j^i) : j \in [m]\} = \{H(\eta', \nu_j^i) : j \in [m]\},$$
$$\{H(\mu, \nu_j^i) : j \in [m]\} \neq \{H(\eta, \nu_j^i) : j \in [m]\}.$$

If we can find the appropriate strings for a clause with 3 positive literals, then we can adapt these to make strings for clauses with negative literals as explained in the last few paragraphs of Section 2.1.1. For the remainder of this appendix, we restrict our attention to a clause c_i with three positive literals.

For $\nu_j^i \in \mathcal{C}_i$, define

$$\nu_j^i[S_i] := (\nu_j^i[x_\alpha], \nu_j^i[y_\alpha], \nu_j^i[x_\beta], \nu_j^i[y_\beta], \nu_j^i[x_\gamma], \nu_j^i[y_\gamma]).$$

For a median $\mu \in \mathcal{M}'(\nu_1^i, \ldots, \nu_m^i)$, define $\mu[S_i]$ similarly.

We say that two strings η and $\overline{\eta}$ are *complementary on the first* $2n$ *coordinates* if $\eta[x_i] = 1 - \overline{\eta}[x_i]$ and $\eta[y_i] = 1 - \overline{\eta}[y_i]$ for each $i \in [n]$. To achieve our first goal, strings will be added to $\mathcal{C}_i$ in pairs which are complementary on the first $2n$ coordinates. Thus x_j and y_j will be ambiguous coordinates for all $j \in [n]$. The e_j coordinates will be for additional ones, so $\mu[e_j] = 0$ for all $\mu \in \mathcal{M}$ and all $j \in [t]$. Therefore $\mathcal{M} = \{0,1\}^{2n} \times \{0\}^t$.

For any $\mu \in \mathcal{M}'$, $\mu[x_j] \neq \mu[y_j]$ for each $j \in [n]$, so $\mu[S_i] \in \{01, 10\}^3$. The eight possible 6-bit strings can be visualized on the 3-dimensional cube. Label the vertices with these strings so that the Hamming distance between two strings is precisely twice the graph distance between the vertices they label. This is represented in Figure A.1. We often use the representation on the right in Figure A.1.

In Definition 2.1, we defined a bijection between $\mathcal{M}'$ and truth assignments. The clause c_i is not satisfied if all of its variables are false. Since c_i has 3 positive literals, this truth assignment corresponds to $\mu \in \mathcal{M}'$ with $\mu[S_i] = 010101$. But if $\mu[S_i]$ is any of the other 7 tuples then the corresponding truth assignment satisfies c_i. Let u_f be the vertex labeled 010101.

Using u_f as a reference point, we say that the *height* of a vertex is its graph distance from u_f. In particular, the vertices labeled $100101, 011001$, or 010110 have height 1. We will also refer to the medians $\mu \in \mathcal{M}'$ with $\mu[S_i]$ equal to one of these as having height 1. The vertices labeled $101001, 100110, 011010$ have height 2 and the medians $\mu \in \mathcal{M}'$ with $\mu[S_i]$ equal to one of these three 6-bit strings have height 2 . Medians $\mu \in \mathcal{M}'$ with $\mu[S_i] = 101010$ have height 3 while medians $\mu \in \mathcal{M}'$ with $\mu[S_i] = 010101$ have height 0.

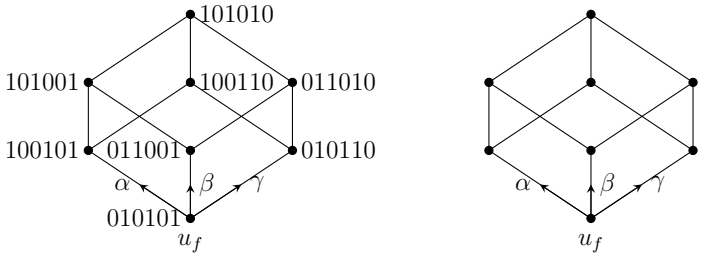


Figure A.1 A labeling of the 3-dimensional cube with the possible values of $\mu[S_i]$. The vertex u_f will correspond to the medians μ with $\mu[S_i] = 010101$. We use the figure on the right to represent the cube.

Next we work toward the second goal of distinguishing the multiset of Hamming distances for medians in $\mathcal{M}'_{c_i}$ from the multiset for medians in $\mathcal{M}' \setminus \mathcal{M}'_{c_i}$. To simplify, we make the restriction that each $\nu_j^i \in \mathcal{C}_i$ will be taken from the collection $S' \uplus \overline{S'}$ where

$$S' = \{\eta \in S : \eta[x_\ell] = \eta[y_\ell] = 0 \ \ \forall \ell \in [n] \setminus \{\alpha, \beta, \gamma\}\},$$

$$\overline{S'} = \{\eta \in S : \eta[x_\ell] = \eta[y_\ell] = 1 \ \ \forall \ell \in [n] \setminus \{\alpha, \beta, \gamma\}\}.$$

For $\mu \in \mathcal{M}'$ and $\nu_j^i \in S' \uplus \overline{S'}$, for each $\ell \in [n] \setminus \{\alpha, \beta, \gamma\}$,

$$H((\nu_j^i[x_\ell], \nu_j^i[y_\ell]), (\mu[x_\ell], \mu[y_\ell])) = 1.$$

This is because $\nu_j^i[x_\ell] \neq \nu_j^i[y_\ell])$ while $\mu[x_\ell] = \mu[y_\ell]$. Consequently, if ν_j^i has $e(\nu_j^i)$ additional ones, then

$$H(\nu_j^i, \mu) = (n-3) + H(\nu_j^i[S_i], \mu[S_i]) + e(\nu_j^i).$$

Each $\eta \in S$ has a string $\eta' \in \overline{S'}$ which is complementary to it on the first $2n$ coordinates. (The choice of η' is not unique.) Following the decision that the strings in $\mathcal{C}_i$ will be in pairs which are complementary on the first $2n$ coordinates, we will select our strings from S' for $\mathcal{C}_i$ and then include a corresponding string from $\overline{S'}$.

Now we analyze the strings from S' to inform our decision on which ones to include in $\mathcal{C}_i$. In order to characterize a string $\nu_j^i \in S'$, we only need to specify $\nu_j^i[S_i]$ and the

Table A.1 Hamming distances when $\eta \in N_1$ with $\eta[x_\alpha] = 0$ and $\eta[y_\alpha] = 1$.

$\mu[S_i]$	$H(\eta[S_i], \mu[S_i])$
010101	2
100101	4
011001	2
010110	2
101001	4
100110	4
011010	2
101010	4

number of additional ones it will have (see Definition 2.4). Aside from the number of additional ones in each string, there are only 2^6 possible choices for $\nu_j^i[S_i]$.

Partition S' into sets N_0, N_1, N_2, N_3 defined here. For each $i \in \{0, 1, 2, 3\}$ and $\eta \in S'$, $\eta \in N_i$ precisely when

$$|\{j : j \in \{\alpha, \beta, \gamma\} \text{ and } \eta[x_j] \neq \eta[y_j]\}| = i.$$

We will consider the sets N_0, N_1, N_2, N_3 individually.

Set N_0: For $\xi \in N_0$ and $\mu \in \mathcal{M}'$, $H(\xi[S_i], \mu[S_i]) = 3$. This is because $\mu[x_j] \neq \mu[y_j]$ while $\xi[x_j] = \xi[y_j]$ for each $j \in \{\alpha, \beta, \gamma\}$. Because no distinction is made between the median in $\mathcal{M}'$, the strings in N_0 will not be useful in accomplishing our second goal.

Set N_1: For $\eta \in N_1$ and $\mu \in \mathcal{M}'$, $H(\eta[S_i], \mu[S_i]) \in \{2, 4\}$. In particular, if $\eta[x_\alpha]$ does not equal $\eta[y_\alpha]$, then $H((\mu[x_j], \mu[y_j]), (\eta[x_j], \eta[y_j])) = 1$ for $j \in \{\beta, \gamma\}$ and either

- $\mu[x_\alpha] = \eta[x_\alpha]$ and $\mu[y_\alpha] = \eta[y_\alpha]$ in which case $H(\eta[S_i], \mu[S_i]) = 2$, or
- $\mu[x_\alpha] \neq \eta[x_\alpha]$ and $\mu[y_\alpha] \neq \eta[y_\alpha]$ in which case $H(\eta[S_i], \mu[S_i]) = 4$.

The values are detailed in Table A.1.

We can display these Hamming distances on the cube representation of medians so that the vertex representing $\mu[S_i]$ is labeled $H(\mu[S_i], \eta[S_i])$. For each $j \in \{\alpha, \beta, \gamma\}$,

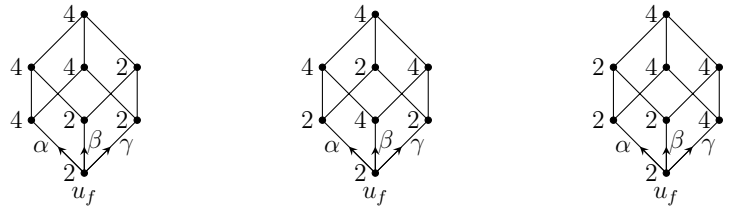


Figure A.2 For $j \in \{\alpha, \beta, \gamma\}$, the values of $H(\eta_j[S_i], \mu[S_i])$ are displayed on the cube representation of medians from Figure A.1 .

let $\eta_j \in N_1$ be the strings with $\eta_j[x_j] = 0$ and $\eta_j[y_j] = 1$ and no additional ones. Set

$$\mathcal{N}_1 := \{\eta_\alpha, \eta_\beta, \eta_\gamma\}.$$

There are three cubes drawn in Figure A.2, one for each of the strings in $\mathcal{N}_1$ with vertices labeled by the value of $H(\mu[S_i], \eta[S_i])$ for $\eta \in \mathcal{N}_1$. The left-most cube corresponds directly to Table A.1, the value of $H(\mu[S_i], \eta_\alpha[S_i])$ labeling the vertices.

Consider the multiset

$$H_1(\mu) := \{H(\mu[S_i], \eta_\alpha[S_i]), H(\mu[S_i], \eta_\beta[S_i]), H(\mu[S_i], \eta_\gamma[S_i])\}.$$

Notice that all medians with height one have $H_1(\mu) = \{2, 2, 4\}$ and all height two medians have $H_1(\mu) = \{2, 4, 4\}$. We can display these values on the median cube. Since vertices of the same height have the same value for H_1, we write each multiset only once. The cube is drawn in Figure A.3.

For each $\eta \in \mathcal{N}_1$, define $\overline{\eta}$ to be the binary string which is complementary to η on the first $2n$ coordinates and has no additional ones. Set

$$\overline{\mathcal{N}_1} := \{\overline{\eta_\alpha}, \overline{\eta_\beta}, \overline{\eta_\gamma}\}.$$

For $\overline{H_1}(\mu) = \{H(\mu[S_i], \overline{\eta_\alpha}[S_i]), H(\mu[S_i], \overline{\eta_\beta}[S_i]), H(\mu[S_i], \overline{\eta_\gamma}[S_i])\}$, we label the cube on the right in Figure A.3.

Notice, however, for all $\mu \in \mathcal{M}'$,

$$H_1 \uplus \overline{H_1} = \{H(\mu[S_i], \eta[S_i]) : \eta \in \mathcal{N}_1 \cup \overline{\mathcal{N}_1}\} = \{2, 2, 2, 4, 4, 4\}.$$

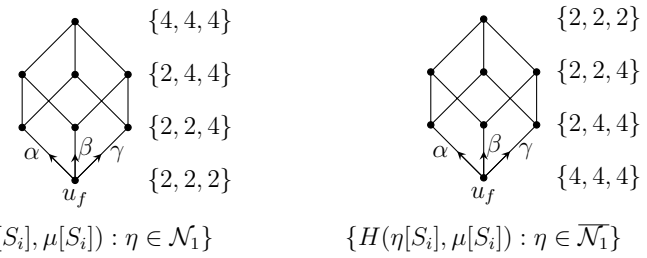


Figure A.3 The left cube displays the values of $H_1(\mu)$ and the right cube displays the values of $\overline{H_1}(\mu)$.

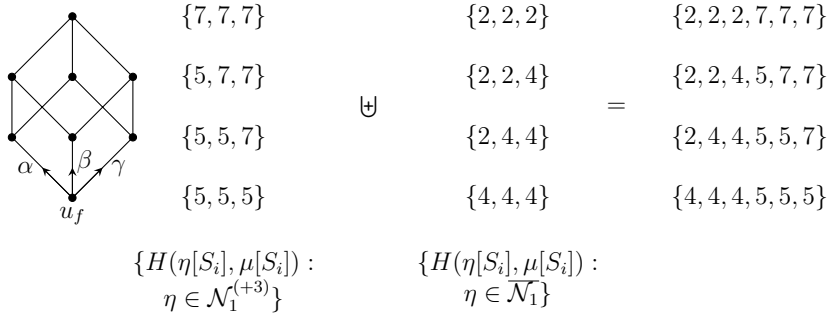


Figure A.4 The values $\left\{H(\eta[S_i], \mu[S_i]) : \eta \in \mathcal{N}_1^{(+3)}\right\} \uplus \overline{H_1}(\mu)$ displayed on the median cube.

So $\mathcal{N}_1 \cup \overline{\mathcal{N}_1}$ acts like an identity.

Now if we give some additional ones to each of $\eta_\alpha, \eta_\beta, \eta_\gamma$, then we increase the Hamming distances by that amount and obtain a more useful collection. For example, if we add 3 additional ones to each string in $\mathcal{N}_1$ to make the set $\mathcal{N}_1^{(+3)} = \{\eta_\alpha^{(+3)}, \eta_\beta^{(+3)}, \eta_\gamma^{(+3)}\}$.

Recall that for any $\mu \in \mathcal{M}'$ and $\eta \in S'$, with e being the number of additional ones in ν,

$$H(\mu, \nu) = H(\mu[S_i], \nu[S_i]) + e(\nu) + (n - 3).$$

Therefore $H(\eta_\alpha^{+3}[S_i], \mu[S_i]) = H(\eta_\alpha[S_i], \mu[S_i]) + 3$. The values of the multiset

$$\left\{H(\eta[S_i], \mu[S_i]) : \eta \in \mathcal{N}_1^{(+3)}\right\} \uplus \left\{H(\eta[S_i], \mu[S_i]) : \eta \in \overline{\mathcal{N}_1}\right\}$$

are displayed in Figure A.4.

Set N_2: Now we turn our attention to the set N_2. For $\zeta \in N_2$ and $\mu \in \mathcal{M}'$,

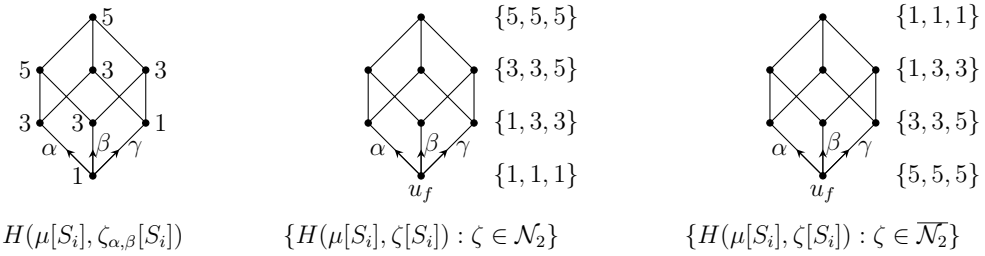


Figure A.5 The left cube displays the Hamming distances $H(\mu[S_i], \zeta_{\alpha,\beta}[S_i])$. The middle gives $H_2(\mu)$ and the right cube displays $\overline{H_2}(\mu)$.

$H(\zeta[S_i], \mu[S_i]) \in \{1, 3, 5\}$. For example, if ζ is a string in N_2 with $\zeta[x_\alpha] \neq \zeta[y_\alpha]$ and $\zeta[x_\beta] \neq \zeta[y_\beta]$, then the following hold:

- If $\mu[x_\alpha] = \zeta[x_\alpha]$ and $\mu[x_\beta] = \zeta[x_\beta]$, then $H(\zeta[S_i], \mu[S_i]) = 1$.
- If $\mu[x_\alpha] = \zeta[x_\alpha]$ and $\mu[x_\beta] \neq \zeta[x_\beta]$, then $H(\zeta[S_i], \mu[S_i]) = 3$.
- If $\mu[x_\alpha] \neq \zeta[x_\alpha]$ and $\mu[x_\beta] = \zeta[x_\beta]$, then $H(\zeta[S_i], \mu[S_i]) = 3$.
- If $\mu[x_\alpha] \neq \zeta[x_\alpha]$ and $\mu[x_\beta] \neq \zeta[x_\beta]$, then $H(\zeta[S_i], \mu[S_i]) = 5$.

For each pair $\{j, \ell\} \subset \{\alpha, \beta, \gamma\}$, $j \neq \ell$, let $\zeta_{j,\ell}$ be a string in N_2 which satisfies $\zeta_{j,\ell}[x_j] = 0 = 1 - \zeta_{j,\ell}[y_j]$ and $\zeta_{j,\ell}[x_\ell] = 0 = 1 - \zeta_{j,\ell}[y_\ell]$ and no additional ones. Let

$$\mathcal{N}_2 = \{\zeta_{\alpha,\beta}, \zeta_{\alpha,\gamma}, \zeta_{\beta,\gamma}\}.$$

The Hamming distances of medians with these three strings are displayed on the three cubes in Figure A.5.

Define the multiset $H_2(\mu) = \{H(\zeta[S_i], \mu[S_i]) : \zeta \in \mathcal{N}_2\}$. The medians at height 1 on the cube all have $H_2(\mu) = \{1, 3, 3\}$ and the medians at height 2 on the cube all have $H_2(\mu) = \{3, 3, 5\}$. This is displayed on the middle cube in Figure A.5. For each $\{j, \ell\} \subset \{\alpha, \beta, \gamma\}$, define $\overline{\eta_{j,\ell}}$ to be the binary string which is complementary to μ on the first $2n$ coordinates and has no additional ones. The right cube of Figure A.5 displays the Hamming distances between $\mu[S_i]$ and the strings in

$$\overline{\mathcal{N}_2} = \left\{\overline{\zeta_{\alpha,\beta}}, \overline{\zeta_{\alpha,\gamma}}, \overline{\zeta_{\beta,\gamma}}\right\}.$$

Table A.2 The 8 different sets of 3 strings in N_2 and their Hamming distances with medians in $\mathcal{M}'$.

$\mu[S_i]$ $\setminus u_f$	101010	101001	100110	011010	100101	011001	010110	010101
101010	$\{1,1,1\}$	$\{1,3,3\}$	$\{1,3,3\}$	$\{1,3,3\}$	$\{3,3,5\}$	$\{3,3,5\}$	$\{3,3,5\}$	$\{5,5,5\}$
101001	$\{1,3,3\}$	$\{1,1,1\}$	$\{3,3,5\}$	$\{3,3,5\}$	$\{1,3,3\}$	$\{1,3,3\}$	$\{5,5,5\}$	$\{3,3,5\}$
100110	$\{1,3,3\}$	$\{3,3,5\}$	$\{1,1,1\}$	$\{1,3,3\}$	$\{1,3,3\}$	$\{5,5,5\}$	$\{1,3,3\}$	$\{3,3,5\}$
011010	$\{1,3,3\}$	$\{3,3,5\}$	$\{3,3,5\}$	$\{1,1,1\}$	$\{5,5,5\}$	$\{1,3,3\}$	$\{1,3,3\}$	$\{3,3,5\}$
100101	$\{3,3,5\}$	$\{1,3,3\}$	$\{1,3,3\}$	$\{5,5,5\}$	$\{1,1,1\}$	$\{3,3,5\}$	$\{3,3,5\}$	$\{1,3,3\}$
011001	$\{3,3,5\}$	$\{1,3,3\}$	$\{5,5,5\}$	$\{1,3,3\}$	$\{3,3,5\}$	$\{1,1,1\}$	$\{3,3,5\}$	$\{1,3,3\}$
010110	$\{3,3,5\}$	$\{5,5,5\}$	$\{1,3,3\}$	$\{3,3,5\}$	$\{3,3,5\}$	$\{3,3,5\}$	$\{1,1,1\}$	$\{1,3,3\}$
010101	$\{5,5,5\}$	$\{3,3,5\}$	$\{3,3,5\}$	$\{3,3,5\}$	$\{1,3,3\}$	$\{1,3,3\}$	$\{1,3,3\}$	$\{1,1,1\}$

The left-most column lists the 8 possible strings $\mu[S_i]$ for $\mu \in \mathcal{M}'$. In the column headers, the 6-bit string is the vertex which acts like u_f in the corresponding rotation of the middle cube in Figure A.5. The entry in row j, column ℓ, is the multiset of Hamming distances between $\mu[S_i]$ and each of the three strings in N_2 corresponding to the cube rotation. For example, the column with heading 101010 corresponds to the right cube in Figure A.5.

This time, $H_1(\mu) \uplus \overline{H_1}(\mu)$ does not result in the same value for all $\mu \in \mathcal{M}'$ so we do not have an identity. Instead, we require 8 different rotations of the middle cube in Figure A.5 to make an identity. The right-most cube is one rotation with the 101010 vertex acting like u_f. Figure A.2 lists the 8 rotations. Each column corresponds to a rotation with the column heading telling the label of the vertex from Figure A.1 which acts like u_f. There is a row for each $\mu[S_i]$, $\mu \in \mathcal{M}'$. The entries in the table indicate the multiset of Hamming distances with $\mu[S_i]$ in the corresponding cube rotation. For example, the column with heading 101010 corresponds to the right-most cube in Figure A.5. The column with heading 010101 corresponds to the middle cube in Figure A.5. If we use all 24 strings (3 strings from each of the 8 rotations), the union of the Hamming distance multisets in a single row of Figure A.2 is $\{1^6, 3^{12}, 5^6\}$ for any row. This multiset $\mathcal{I}_2$ of 24 strings is an identity since it does not distinguish medians in $\mathcal{M}'$.

Set N_3: Let τ be a binary string in N_3. Then for any $\mu \in \mathcal{M}'$, the Hamming distance $H(\mu[S_i], \tau[S_i])$ will be in $\{0, 2, 4, 6\}$. This comes from the fact that $H(\mu[S_i], \tau[S_i]) = 2\ell$ whenever $|\{j \in \{\alpha, \beta, \gamma\} : \mu[x_j] \neq \tau[x_j]\}| = \ell$.

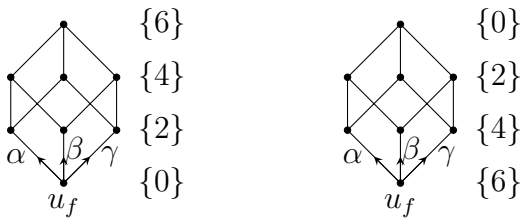


Figure A.6 The Hamming distances between a median on the cube and string τ with $N(\tau) = 3$

Let τ be a string in N_3 with $\tau[x_\alpha] = \tau[x_\beta] = \tau[x_\gamma] = 0$ and no additional ones. All medians of height 1 have $H(\mu[S_i], \tau[S_i]) = 2$ and all medians of height 2 have $H(\mu[S_i], \tau[S_i]) = 4$. This is displayed in Figure A.6. Let

$$\mathcal{N}_3 = \{\tau\} \qquad \overline{\mathcal{N}_3} = \{\overline{\tau}\}.$$

Once again, we will need all 8 rotations of this cube to make an identity. This identity will consist of exactly 8 strings.

We have now defined 6 sets of strings, $\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3$ and their complements. For each $j \in [3]$ and $\mu \in \mathcal{M}'$, define

$$\mathcal{H}_j(\mu) := \{H(\mu[S_i], \nu[S_i]) + e(\nu) : \nu \in \mathcal{N}_j\}.$$

We have already seen that for any $\mu, \mu' \in \mathcal{M}'$, both of height 1, $\mathcal{H}_j(\mu) = \mathcal{H}_j(\mu')$. Also, for any $\mu, \mu' \in \mathcal{M}'$, both of height 2, $\mathcal{H}_j(\mu) = \mathcal{H}_j(\mu')$.

For $j \in [3]$ and non-negative integer e, let $\mathcal{N}_j^{(+e)}$ be the set of strings in $\mathcal{N}_j$ with e additional ones added to each string. The set $\overline{\mathcal{N}_j}^{(+e)}$ is defined similarly.

The goal is to find a multiset $\mathcal{C}_i'$ of $\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3$ and their complements, possibly with additional ones added to each, such that

- there is a one-to-one correspondence between the sets of the form $\mathcal{N}_j^{(+e)}$ and the sets of the form $\overline{\mathcal{N}_j}^{(+e')}$, and

- there is a method to take unions and set subtractions of the multisets

$$\left\{H(\mu[S_i],\nu[S_i]) : \nu \in \mathcal{N}_j^{(+e)}\right\} \uplus \left\{H(\mu[S_i],\nu[S_i]) : \nu \in \mathcal{N}_j^{(+e')}\right\}$$

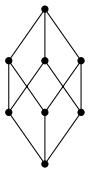
for each $\mathcal{N}_j^{(+e)} \in \mathcal{C}_i'$ so that for any medians $\mu, \mu' \in \mathcal{M}'$ of height 1,2, or 3 on the median cube and $\mu_0 \in \mathcal{M}'$ with $\mu_0[S_i] = 010101$, then

* $\mathcal{H}_j(\mu) = \mathcal{H}_j(\mu')$ and
* $\mathcal{H}_j(\mu) \neq \mathcal{H}_j(\mu_0)$.

One possible multiset is $\mathcal{C}_i' = \left\{\mathcal{N}_1^{(+3)}, \overline{\mathcal{N}_1}^{(+0)}, \mathcal{N}_2^{(+2)}, \overline{\mathcal{N}_2}^{(+1)}, \mathcal{N}_3^{(+1)}, \overline{\mathcal{N}_3}^{(+2)}\right\}$ with

$$\left(\mathcal{N}_1^{(+3)} \cup \overline{\mathcal{N}_1}^{(+0)}\right) \setminus \left(\mathcal{N}_2^{(+2)} \uplus \overline{\mathcal{N}_2}^{(+1)}\right) \uplus \left(\mathcal{N}_3^{(+1)} \uplus \overline{\mathcal{N}_3}^{(+2)}\right).$$

The resulting Hamming distances are displayed in Figure A.7.



$$\begin{array}{ccccccccccc}
\{7,7,7\} & & \{2,2,2\} & & \{7,7,7\} & & \{2,2,2\} & & \{7\} & & \{2\} & & \{2,7\} \\
\{5,7,7\} & \uplus & \{2,2,4\} & \setminus & \{5,5,7\} & \setminus & \{2,4,4\} & \uplus & \{5\} & \uplus & \{4\} & = & \{2,7\} \\
\{5,5,7\} & & \{2,4,4\} & & \{3,5,5\} & & \{4,4,6\} & & \{3\} & & \{6\} & & \{2,7\} \\
\{5,5,5\} & & \{4,4,4\} & & \{3,3,3\} & & \{6,6,6\} & & \{1\} & & \{8\} & & \neq \{2,7\}
\end{array}$$

$$\left(H_1^{(+3)} \uplus \overline{H_1}^{(+0)}\right) \setminus \left(H_2^{(+2)} \uplus \overline{H_2}^{(+1)}\right) \uplus \left(H_3^{(+1)} \uplus \overline{H_3}^{(+2)}\right)$$

u_f

Figure A.7 A selection which distinguishes the Hamming distances at u_f from the rest.

Ultimately, we need a union of strings that will accomplish the same final goal, without the use of set subtractions. However, we can easily modify the collection in Figure A.7 to remove the set subtraction by essentially adding the identities $\mathcal{I}_2^{(+2)}$ and $\overline{\mathcal{I}_2}^{(+1)}$. The first one contains $\mathcal{N}_2^{(+2)}$ as a subset, so $\mathcal{I}_2^{(+2)} \setminus \mathcal{N}_2^{(+2)}$ is a collection of 21 strings. Likewise, $\overline{\mathcal{I}_2}^{(+1)}$ contains $\overline{\mathcal{N}_2}^{(+1)}$.

Let

$$\mathcal{C}_i = \left(\mathcal{N}_1^{(+3)} \uplus \overline{\mathcal{N}_1}^{(+0)}\right) \biguplus \left(\mathcal{N}_3^{(+1)} \uplus \overline{\mathcal{N}_3}^{(+2)}\right) \biguplus \left(\left(\mathcal{I}_2^{(+2)} \setminus \mathcal{N}_2^{(+2)}\right) \uplus \left(\overline{\mathcal{I}_2}^{(+1)} \setminus \overline{\mathcal{N}_2}^{(+1)}\right)\right).$$

This is a collection of 50 strings such that for any $\mu \in \mathcal{M}'$, μ of height at least one in the median cube,

$$\begin{aligned}\{H(\mu[S_i], \nu[S_i]) + e(\nu) : \nu \in \mathcal{C}_i\} &= \{3^6, 5^{12}, 7^6\} \cup \{2^6, 4^{12}, 6^6\} \cup \{2, 7\} \\ &= \{2^7, 3^6, 4^{12}, 5^{12}, 6^6, 7^7\}.\end{aligned}$$

On the other hand, if $\mu \in \mathcal{M}'$ has $\mu[S_i] = 010101$, then

$$\begin{aligned}\{H(\mu[S_i], \nu[S_i]) + e(\nu) : \nu \in \mathcal{C}_i\} &= \{3^6, 5^{12}, 7^6\} \cup \{2^6, 4^{12}, 6^6\} \uplus \{1, 4^3, 5^3, 8\} \setminus \{3^3, 6^3\} \\ &= \{1, 2^6, 3^3, 4^{15}, 5^{15}, 6^3, 7^6, 8\}.\end{aligned}$$

By adding $n - 3$ to each of these values, we obtain the Hamming distances $H(\mu, \nu)$ which are listed in Table 2.1.

Table 3.1 details a different set of 26 strings strings for clause c_i in exactly the same way as they are explained for Table 2.1. The conclusions are also the same.